Smart invaders of private matters

Privacy of communication on the Internet and in the Internet of Things (IoT)

Anna Krasnova

IPA Dissertation Series: 2017-05
ISBN/EAN: 9789462336780
Typeset using LaTeX 2ε (MacTeX-2017)

Based on template by Pim Vullers https://github.com/pimvullers/phd_thesis
Cover design by Yana Frank (Miu Mau)
Cover image is inspired by Space Invaders by the Ltd. Taito
Printed by Gildeprint, Enschede (https://www.gildeprint.nl/en/)

Radboud University

This research was sponsored by by SIDN.nl (http://www.sidn.nl/) and conducted within the Privacy and Identity Lab (PI.lab, http://www.pilab.nl).
The author was employed at the Radboud University Nijmegen.
The work in this thesis has been carried out under the auspices of the research school IPA (Institute for Programming research and Algorithmics).

Privacy of communication on the Internet and in the Internet of Things (IoT)

PROEFSCHRIFT

ter verkrijging van de graad van doctor
aan de Radboud Universiteit Nijmegen
op gezag van de rector magnificus prof. dr. J.H.J.M. van Krieken,
volgens besluit van het college van decanen
in het openbaar te verdedigen op **maandag 9 oktober 2017**
om **14.30 uur** precies

door

Anna Krasnova

geboren op 26 september 1987
in de Sovjet-Unie

Promotor:

Prof. dr. Lejla Batina

Copromotor:

Dr. J.H. Hoepman

Manuscriptcommissie:

Prof. dr. J.J.C. Daemen	
Prof. dr. Gildas Avoine	INSA Rennes, Frankrijk
Prof. dr. ir. Bart Preneel	Katholieke Universiteit Leuven, België
Dr. Carmela Troncoso	IMDEA Software Institute, Spanje
Prof. dr. George Danezis	University College London, Verenigd Koninkrijk

Privacy of communication on the Internet and in the Internet of Things (IoT)

DOCTORAL THESIS

to obtain the degree of doctor
from Radboud University Nijmegen
on the authority of the Rector Magnificus, prof. dr. J.H.J.M. van Krieken,
according to the decision of the Council of Deans
to be defended in public on **Monday, October 9, 2017**
at **14:30 hours**

by

Anna Krasnova

Born on September 26, 1987
in USSR

Supervisor:

Prof. dr. Lejla Batina

Co-supervisor:

Dr. J.H. Hoepman

Doctoral Thesis Committee:

Prof. J.J.C. Daemen	
Prof. dr. Gildas Avoine	INSA Rennes, France
Prof. dr. ir. Bart Preneel	Katholieke Universiteit Leuven, Belgium
Dr. Carmela Troncoso	IMDEA Software Institute, Spain
Prof. dr. George Danezis	University College London, United Kingdom

Acknowledgments

The years I spent working on this thesis were passing by very slowly. It felt as if it would last forever. And yet, now that I am writing this last bit of my work, it seems unbelievable that the time is almost gone. I am starting to miss it already.

First and foremost I want to thank my promotor Lejla Batina. Without her, this thesis would never have been possible, and these are not just formal words. Her support, kindness and willingness to give her best for her PhD students have carried me through the rough period of my PhD life. I would like to thank her for the freedom she gave me, and for everything I learned from her.

Many people provided help and advice during the writing of this thesis. I thank Lejla Batina, Tommy Koens, Gergely Alpar, Peter Schwabe, Robert Furber, Jaap-Henk, Markus Klinik for reading parts of this thesis and giving me feedback and suggestions. I thank Lynn Batten for her advice on the thesis structure and time planning. I thank Sven Kiljan and Joeri de Ruiter for helping me out with translating the abstract to this thesis.

I would also like to also thank Bart Jacobs for his support. I have seen many examples that show he takes seriously the overall atmosphere, as well as needs of everyone in the group. A good illustration of this is the dedication he put into settling the whole group on the same floor after we changed our building, in order to keep tight contacts in the group. This dedication largely contributes to the fact that working at the Digital Security group is a big pleasure!

I would like to thank my colleagues who contributed immensely to my personal and professional time during my PhD. The group has been a source of friendships as well as good advice and collaboration. In particular, Joeri de Ruiter, who worked tirelessly with me on making cMix into a publication. I enjoyed our collaboration and discussions a lot. I am grateful to Gergely Alpár for being an inspiration to me. Special thanks to Peter Schwabe, who gave me extremely useful advice and support throughout my PhD time. I would like to thank Andreas Hülsing for his

generous help with finishing my first protocol proofs. Many thanks to Robert Furber for brightening up my day with his great sense of humor. I am thankful for all the fun moments I shared with Antonio de la Piedra, Pedro Maat Massolino, Louiza Papachristodoulou, Brinda Hampiholi.

My sincere gratitude to Daniel J. Bernstein and Tanja Lange for all their kindness, help and advice. It's been a privilege and fun to work under your supervision on Elligator. It was also a great privilege, as well as an adventure, to work with David Chaum. I would like to thank all my co-authors: Gergely Alpár, Lejla Batina, Lynn Batten, Veelasha Moonsamy, Antoine Guellier, Iynkaran Natgunanathan, Jens Hermans, Jaap-Henk Hoepman, Moritz Neikes, Peter Schwabe, Mike Hamburg, Debajyoti Das, Farid Javani, Aniket Kate, Joeri de Ruiter and Alan T. Sherman.

My special gratitude to Ronny Wichers Schreur and Desiree Hermans. I enjoy living in the same house with you a lot. Thank you for our walks, dinners, and tea breaks. Thank you for being supportive, for your friendship. Desiree, thank you for saving my life. Its hard to even begin to describe how much I owe to you.

A very special group of people are my dearest friends. Sven Kiljan, thank you for being an extremely kind and attentive person with whom I also happen to share a passion for games. Thank you for being there for me when I needed it most. I would like to thank both you and your girlfriend Manxia Liu for all the great time we spent together. Tommy Koens, I do not even know where to begin to thank you. Thank you for inspiring me, for the humor and insights you shared with me. I treasure our Thursday walks and discussions. Dmitrij Zjazin, thank you for being my good friend for many years. Thank you for our, at times heated, discussions, and all the fun moments we shared together.

I am immensely grateful to Andreas Pfitzmann for inspiring me to start my study in Germany, and to start a PhD after that. For supporting a travel to my very first conference when I was still a Master student, and for asking me to give my first rump session talk there, which was completely unexpected. I am most grateful for the chance I had to work with him. Even laying in a hospital bed, before his last operation, he stayed the very same person I always experienced him: attentive to people around him, driven by his work and ideas. His death was one of the saddest events in my life.

I would like to mention here the people most essential and special to me. My grandma Lida, my grandma Elda, my uncle Misha, my father, and my mum. Thank you for all your love. Mum, thank you for being not only a mum, but a close friend. I enjoy our shared adventures the most.

Samenvatting

Deze thesis stelt technologische gereedschappen voor die privacy waarborgen voor communicatie op het Internet en in het Internet of Things (IoT). IoT werd decennia geleden voorgesteld als een alomtegenwoordig netwerk, bestaande uit alledaagse dingen die informatie over hun omgeving kunnen waarnemen, wijzigen en verwerken. Voor IoT wordt een hoge mate van heterogeniteit verondersteld van deelnemende apparaten en netwerken. IoT apparaten kunnen qua rekenkracht en energieverbruik zwaar of licht zijn, en deze kunnen een vaste locatie hebben of mobiel zijn met een potentieel instabiele verbinding. Netwerken van IoT apparaten kunnen variëren van netwerken waarbij de apparaten fysiek dicht bij elkaar zijn opgesteld tot netwerken waarbij de afstand tussen apparaten aanzienlijk groter is.

De huidige IoT oplossingen verschillen van hoe ze werden voorgesteld. Eén van de belangrijkste verschillen is de afhankelijkheid van het Internet als enige netwerk. Een typisch IoT consumentenapparaat is vandaag de dag verbonden met een backend via het Internet, wat meestal een IoT cloud is. Bediening op afstand van het apparaat is vaak mogelijk via een ander apparaat dat is verbonden met dezelfde cloud. Zonder deze verbinding bieden IoT apparaten beperkte of geen functionaliteit. Hierom betrekken we ook het Internet bij onze discussies over privacy in het Internet of Things.

In veel scenario's is het beschermen van enkel de inhoud van de communicatie niet voldoende. De identiteit van de communicerende partijen dient ook beschermd te worden. Voor communicatie via het Internet is voor bijna alle gebruiksdoelen informatie nodig die gebruikt kan worden om de identiteit van gebruikers te onthullen. Een voorbeeld hiervan is de onthulling van netwerkadressen van zenders en ontvangers, wat hen kwetsbaar maakt voor identificatie en tracking. Transacties via het Internet hebben vaak unieke persoonsgegevens nodig van gebruikers voor verschillende functies, zoals authenticatie, autorisatie en het filteren van infor-

matie. Transacties die gekoppeld zijn aan dezelfde persoonsgegevens zijn te volgen. Hierdoor zijn gebruikers te volgen en wordt hun privacy bedreigd.

Deze thesis verkent communicatieprotocollen die privacy bieden op het Internet en in netwerken van apparaten die dicht bij elkaar geplaatst zijn. Besproken apparaten variëren in rekenkracht en energieverbruik, van lichtgewicht tot zwaargewicht. Voor verschillende soorten netwerken en apparaten verbeteren of introduceren we concrete protocollen die de privacy beschermen door het beperkt vrijgeven van informatie door communicatieprotocollen en applicaties.

Hoofdstuk 3 is toegewijd aan het bestuderen van de transacties van gebruikers in het IoT, met de nadruk op verzoeken vanaf IoT gebruikersapparaten naar diensten waarvoor de gebruiker is geregistreerd. Attribute-based (AB) authenticatie zorgt voor privacy door te voorkomen dat transacties met elkaar kunnen worden geassocieerd. Ook wordt in dit hoofdstuk aangetoond dat AB authenticatie gebruikers controle geeft over hun persoonsgegevens, en dat het zorgt voor gegevensminimalisatie en doelbeperking, twee belangrijke principes voor gegevensbescherming

Hoofdstuk 4 richt zich op privacy-vriendelijke protocollen die overeenkomende attributen kunnen herkennen tussen twee RFID tags en een RFID lezer. Deze protocollen werken zonder dat zij waardes van de attributen prijsgeven aan de lezer of andere RFID tags. De omschreven protocollen kunnen meerdere attributen vergelijken per tag, waardoor hun toepassingsgebied wordt verbreed. Daarnaast zijn de protocollen zeer efficiënt met betrekking tot de benodigde rekenkracht. Dit geldt in het bijzonder voor één van de omschreven protocollen, die enkel een lichtgewicht hashfunctie vereist op de tags. Twee andere protocollen hebben daarnaast ook asymmetrische encryptie nodig, wat mogelijk is met krachtigere tags.

Anonieme communicatieprotocollen worden onderzocht in hoofdstuk 5. Een voorstel wordt gegeven om een anoniem planningsprobleem op te lossen voor het Dining Cryptographers protocol, wat is voorgesteld door Chaum in 1988. Dit protocol zorgt voor sterke waarborgen voor anonimiteit, maar is beperkt in snelheid en schaalbaarheid, en kent verschillende problemen die praktische implementaties complex maken. In dit hoofdstuk wordt een specifiek probleem opgelost, namelijk het reserveren van slots voor communicatie. Het gebruik van *footprint scheduling* wordt voorgesteld als een nieuwe techniek voor deelnemers om communicatieslots te onderhandelen zonder verlies van anonimiteit. Daarnaast blijft het aantal actieve deelnemers onbekend. Footprint scheduling is simpel, efficiënt, en het resultaat is

zeer goed, in het bijzonder in zeer dynamische netwerken waarbij de deelnemers en de hoeveelheid activiteit regelmatig veranderen.

Hoofdstuk 6 introduceert een nieuwe methode genaamd Elligator, dat zich richt op het voorkomen van censuur. Elligator verbergt het uitwisselen van cryptografische sleutels tussen communicerende partijen. Dit voorkomt het onderscheiden en blokkeren van gebruikers op basis van het soort protocol dat zij gebruiken. Zulke blokkades kunnen zich richten op anonieme communicatiesystemen, bijvoorbeeld wanneer deze systemen verboden zijn of gebruikt worden om censuur te omzeilen. Elligator maakt patronen in de verstuurde informatie bij het uitwisselen van sleutels onherkenbaar, en richt zich in het bijzonder op elliptische kromme punten. Deze punten worden verstuurd bij het uitwisselen van cryptografische sleutels in elliptische kromme cryptografie.

In hoofdstuk 7 wordt een nieuwe versie van een alombekende benadering voor anonieme communicatie gepresenteerd. Dit heeft betrekking op mix-nets, welke sterke bescherming bieden. Het nieuwe protocol maakt mix-nets bruikbaar voor lichtgewicht apparaten binnen het IoT. Het LightMix protocol vereist enkel dat clients eenmalig asymmetrische cryptografische operaties uitvoeren tijdens registratie. Door het vooraf uitvoeren van berekeningen zijn real-time asymmetrische cryptografische operaties niet meer nodig voor zowel zenders, ontvangers als mixnodes. Hierdoor is er minder vertraging als gevolg van cryptografische berekeningen en is er minder rekenkracht vereist bij de clients. De basis real-time fase voert alleen enkele snel te berekenen modulaire vermenigvuldigingen uit. Deze eigenschappen maken LightMix bruikbaar voor applicaties die afhankelijk zijn van korte doorlooptijden op lichtgewicht apparaten terwijl de uitstekende garanties van mix-nets met betrekking tot privacy behouden blijven, waaronder het ontbreken van de koppeling tussen zenders en ontvangers, en de bescherming tegen veel aanvallen op basis van verkeersanalyse.

Summary

This thesis presents technological tools for ensuring privacy of communication on the Internet and in the Internet of Things (IoT). IoT was envisioned decades ago as a ubiquitous network comprising everyday things that can sense, change, and process information about their environment. IoT promises to be highly heterogeneous in terms of devices and networks. Devices in IoT can be powerful or lightweight in terms of computation ability and available power; they can also be stationary or mobile with potentially unstable connections. Networks comprising IoT can vary from networks of devices located in closest proximity to each other to networks of devices located at large distances.

Existing IoT solutions differ from those originally envisioned. One of the important differences is their heavy reliance on a single network, namely the Internet. One can argue, it is still not an IoT, but rather an Internet with occasional things connected to it. A typical IoT consumer product nowadays is an appliance or a gadget connected via the Internet to a backend, usually an IoT cloud. Often an IoT device can be remotely controlled though another gadget connected to the same cloud. Without this connection, current IoT devices offer either very limited functionality, or none at all. For this reason we also consider the Internet when talking about privacy in the Internet of Things.

In many communication scenarios it is not sufficient to protect only the content of the communication; it is also necessary to protect meta-data, especially meta-data related to the identity of communicating parties. Communication on the Internet requires information that can be used to reveal the identity of users in order to perform almost any function. For example, Internet communication protocols reveal the network addresses of both senders and receivers, making them vulnerable to identification and tracking. Transactions that are performed via the Internet often require identifiers and identifying information about users for a wide range of functions such as authentication, authorization, and information filtering. Trans-

actions linked to the same identifier are traceable, and ultimately also make users traceable; hence their privacy is threatened.

This thesis explores protocols for providing communication privacy both on the Internet and in networks of devices located in close proximity to each other. We consider a range of connected devices, from lightweight in terms of computational power and available power to computationally powerful ones. For different network and device types we improve and design concrete protocols that provide privacy protection by minimizing the information revealed either by communication protocols or by applications.

In Chapter 3 we study users' transactions in the IoT, in particular requests made from users' devices to the services to which these users are subscribed. We apply attribute-based (AB) authentication to provide privacy by ensuring unlinkability between these transactions. In addition, we demonstrate that the use of AB authentication provides users with control over their personal data and achieves data minimization and purpose limitation, both important principles in information privacy.

Chapter 4 considers privacy-friendly protocols that detect matching attributes between two RFID tags and a reader. These protocols succeed without revealing attribute values to either RFID reader or another RFID tag. The protocols presented can perform attribute matching to multiple attributes per tag to broaden the range of possible applications of the protocols. They are also very efficient in terms of computation. Specifically, one of the protocols presented in this chapter only needs a lightweight hash function implemented on the tags. Two other protocols additionally need asymmetric encryption, which is feasible with more powerful tags.

Chapter 5 examines anonymous communication protocols. In particular, it proposes a solution to the anonymous scheduling problem of the Dining Cryptographers protocol proposed by Chaum in 1988. This protocol provides strong anonymity guarantees, however they come at the price of limited performance and scalability and multiple issues that make deployment complicated in practice. In this chapter we address one of those issues, namely slot reservation. We propose *footprint scheduling* as a new technique for participants to negotiate communication slots without losing anonymity and at the same time hiding the number of actively sending users. Footprint scheduling is at the same time simple and efficient and it yields excellent results, in particular in very dynamic networks with a frequently changing set of participants and frequently changing activity rate.

Chapter 6 introduces a tool called Elligator, which deals with censorship prevention. Elligator hides the fact that communicating parties perform cryptographic key exchange. This prevents differentiating and blocking users based on the type of protocol they use. For example, such blocking can happen to anonymous communication systems, when they are forbidden or used as a censorship-circumvention tool. Elligator makes patterns in the information transmitted during key exchange unrecognizable, in particular elliptic curve points. These points are transmitted during a key exchange in elliptic curve cryptography.

Chapter 7 presents a new version of a well-known approach to providing anonymous communication. This involves the use of mix-nets, which guarantees strong protections. The new protocol makes mix-nets suitable for lightweight devices within IoT. The core LightMix protocol requires clients to perform public-key operations only once, during registration. Through precomputation, all real-time public-key operations are eliminated—at the senders, recipients, and mixnodes—thus decreasing cryptographic latency and reducing computational costs for clients. The core real-time phase performs only a few fast modular multiplications. These properties make LightMix suitable for low-latency applications on lightweight devices, while retaining the excellent privacy guarantees of mix-nets, including unlinkability of senders and receivers, and resistance to many traffic-analysis attacks.

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

> *During a meeting people started to argue about what the worst thing in the world is. Some named illnesses, some death, some poverty... Many things were named. Finally, Nasreddin Hodja was asked about his opinion. Nasreddin answered: 'It is bad when what you want doesn't happen'. After a brief consideration he added: 'Though it is much worse when what you don't want happens'.*
>
> - Folklore

Online communication. Online communication has become one of the most significant forms of human communication. Everywhere – at home, in public transport, at a party, even during social dinners in a restaurant – people are sending messages via SMS, Whatsapp, Signal, Telegram, and Facebook. Online communication is also an integral part of services that businesses provide. Online shopping has become so dominant that physical stores are now often referred to as offline stores. Crowds in the shopping malls right before Christmas Eve morph into endless lines in post offices, where people collect their online purchases. One can perform payments, play games, watch television, collaborate, order governmental services, and even fill out tax forms online. It is impossible to disagree with the statement that online communication has become a crucial part of everyday life.

The popularity of online communication came hand in hand with the growing popularity of Big Data. Big data is often used by large companies **??** to understand, predict and shape behavior, which has made data about people one of the most important products of modern businesses. Certain companies such as Towerdata

make a profit solely by collecting, aggregating and selling demographic information about people. Interested buyers are those who hope to increase their sales via, for example, better targeted advertisements, web-site designs or price policies. Demographic information is often collected online via tracking and analyzing user's websurfing behavior, forum posts and even posted pictures.

The perfect example of how much companies can learn about people based on seemingly innocent data is shopping in supermarkets. A book by Duhigg [117] provides a detailed illustration based on an interview with one of the data analysts working at Target, a large supermarket chain. There he describes how information is collected when customers use the supermarket's credit card or a customer card for purchases. This information can be used to learn details about their private lives such as whether they have kids, or a backyard. More intimate things can be learned as well, for example whether they expect a child, the trimester of pregnancy and even the delivery date within a small time window. Duhigg writes: "when someone suddenly starts buying lots of scent-free soap and cotton balls, in addition to hand sanitizers and an astounding number of washcloths, all at once, a few months after buying lotions and magnesium and zinc, it signals they are getting close to their delivery date." A famous scandal that happened several years ago serves as a confirmation. In 2012 it was revealed that Target learned about a teenager's pregnancy earlier than her parents did [315]. Life changing events such as a child birth are valuable for companies, as they are the turning points around which new buying habits can be implanted [117].

Such a situation is not advantageous from the perspective of one's privacy. The popular phrase "privacy is dead" became widely used after the talk "Privacy is dead, get over it" given by private investigator Steve Rambam [266] in 2006. His main message is that given the publicly available information and the information collected at semi-private databases, *very few information items are enough to completely identify almost anybody*. Seven years earlier, in 1999, at that time Sun Microsystems' CEO Scott McNealy made another well-known statement: "You have zero privacy anyway. Get over it.." This was the year when GPS phones just started to appear in shops [295]. Facebook did not exist back in 1999. There was no YouTube or Twitter yet, and Google started this year with only eight employees [158]. In 1999 this statement caused raised eyebrows. Today, especially after Edward Snowden's revelations, "privacy is dead" is the topic of numerous public debates.

At the same time the benefits of online communication facilitated by the Internet have in a certain sense become an ideology. Many people seem to agree that

it improves businesses, people's productivity, collaboration, and gives easy access to any information one might need. Artists, writers, scientists, and office workers are strongly encouraged to be present online in LinkedIn, Facebook, Twitter, etc. or to maintain a blog. A significant number of commercial applications started to push user's data to a cloud for synchronization purposes or even switched to being purely web based. More and more free online services such as email servers, task tracking and collaboration platforms have emerged. Numerous gadgets connected to the Internet, such as smart watches, are designed to save time, track and support daily activities for improved productivity and convenience.

The Internet as a part of the Internet of Things. Thanks to connected gadgets and other things the Internet became a part of a larger network – an Internet of Things (IoT). IoT was envisioned back in 1999 [321] and promised to provide highly sophisticated services with the help of numerous objects in everyday life, things, being interconnected and interoperating. For example, in smart homes a plethora of things can cooperate to make sure that the house owner sleeps as long as possible and still be at work on time. For this, an alarm clock can learn today's weather conditions and request information about traffic jams. This information can be combined with the average times required for travel and today's schedule in order to decide if the user can sleep half an hour longer. His smart watch can influence this decision if it concludes the owner's diet behavior requires him to exercise a bit longer this morning. Then the alarm clock can notify the garage to start warming up the car in time, the coffee maker to start brewing and the fridge to order a fresh bread roll for breakfast.

Many of such examples have been described by futurists. However, IoT is not there yet. There are a multitude of IoT products such as smart meters, fridges, smart watches, and coffee makers connected to the Internet, but they still rarely interoperate to provide the promised sophisticated services. Additionally, the Internet still plays a major role in IoT, compared to the expected heterogeneity of the types of networks within IoT – from networks comprised of devices located inside or on the surface of a person's body to sensor networks of several farm fields.

Definitions of IoT given by industry mostly focus on things connected to the Internet [274, p. 40]. This can be explained by the fact that IoT products are currently supported by online backend systems such as clouds to store user data and to outsource programmable logic. An Internet connection is also used whenever there is a need to access remote functionality of devices such as switching power on and

off [30, 216]. Without the Internet connection such products will either not function at all, or have limited functionality.

Privacy threats. In this thesis we are considering two of the major privacy threats in IoT. The first is that there is *no control* of when and what kind of data is collected. Data about people is being collected, analyzed and traded on a regular basis. The second threat is that the technologies that are currently being used facilitate *data collection that identifies users*.

Data collection can happen manually, but also automatically by surrounding or owned devices. For example, anyone can be videotaped in the most unpleasant moments of their life, and the video uploaded to YouTube to be watched potentially by billions of people. These situations are already ingrained in the culture by popular songs [290] and cartoons [312]. Gadgets we own have 'backdoors' [157, 16] and are reported to collect and send away information about us. The same goes for applications and services. Security cameras are ubiquitous and can recognize people solely by the way they walk [266]. As discussed earlier, such uncontrolled data collection coupled with sophisticated data analysis performed by Big Data analysts can reveal the most intimate details about one's life, as well as make everyone easy to trace.

As more and more gadgets are being invented and things are being connected to the Internet, privacy invasion becomes more and more aggressive. Connected devices appear in previously intimate spaces such as our homes, they are placed on our bodies, and can collect sensitive information about our habits and health. If street cameras and smartphones make it very difficult to relax in public spaces such as streets and offices, appliances in smart homes are likely to make it impossible to have private lives in the very homes we live in [169].

Two major approaches taken in technological solutions support data collection that identifies individuals. The first is the use of centralized architectures, compared to distributed, mesh and peer-to-peer networks. The second is the use of *identifying attributes* (see Section 3.5.2) in communication protocols as well as in transactions within applications built on top of communication protocols. Attributes are qualities or parts of subjects or objects. Identifying attributes are unique for an individual or an object, for example it can be the manufacturing ID of a phone, or a passport number of a citizen. A group of combined attributes can become identifying as well, when it makes it possible to uniquely distinguish a person among other individuals [299]. Such combined attributes are exactly the "very few infor-

mation items [that] are enough to completely identify almost anybody" Rambam was talking about [266].

Identifying attributes. Use of identifying attributes in communication networks started at the very time these networks were invented. Communication networks require addresses (e.g. IP and MAC address) assigned to the nodes in order to route messages to the intended destination. Networking protocols were not designed to consider such addresses as sensitive information. As a result, these addresses are accessible to anyone who can view or intercept communication of users. Knowing network addresses often makes it possible to learn which transactions belong to the same user, and to establish links to physical locations, types of devices used for communication and identities of communicating users.

This way, any action a user is taking when using communication networks is potentially traceable, irrespective of the protocol used for the transaction performed at the application level. Anybody can learn what messages this user sent, to whom, when and for what reason. Consider transferring money with the digital cash system Bitcoin. Bitcoin supports binding money transfers from addresses that are not tied to the identity of users. To create such addresses it is enough to generate new public and private key pairs. Transactions themselves also do not require users to reveal their identity. However, it is possible for a third party to learn who the originator of a money transfer is. For this, it is enough to be able to read an IP address in the messages the user sent during the transaction. An attacker who can oversee a large enough part of the Bitcoin peer-to-peer network can easily perform this task. In a centralized architecture this task would be easy even for a local attacker, who can observe only the incoming and outgoing traffic of a central server.

In order to prevent such an attack, a user has to be able to conceal his IP address, a *communication level* attribute. However, this is not enough to keep him protected. For example, the user's real identity is often tied to his addresses by the wallet or exchange services he uses, despite the protocol not requiring this. In this case his address can be linked, for example, to his name, a *transaction level* attribute. Another way his identity can be revealed is when he performs two money transfers using the same address. Even if his address is not tied to his real name, an adversary will be able to see that both transactions are originating from the same user. If one of these transactions was performed with a store that needed his home address to perform a delivery, this can be used to identify the user in both transactions. Therefore, to protect users' privacy both communication and transaction

level attributes need to be anonymized. More elaboration on the attribute levels considered in this thesis can be found in Section 1.1.

Technological solutions to privacy threats. Privacy-Enhancing Technologies (PETs) are technological tools aimed at protecting the privacy of users, while making it possible to implement useful functionality efficiently. PETs provide users with technical solutions that help to enforce their control over what information is collected about them. PETs are usually designed to achieve control by providing confidentiality and anonymity [160]. Anonymous communication systems are PETs designed to protect communication level attributes. For example, they make it possible to make senders and receivers of messages unknown. Other PETs such as attribute-based credentials (ABCs) anonymize transaction level attributes to protect users' privacy in specific applications.

The positive implication of the Internet dominance within IoT is that many protocols used currently on the Internet can be adopted for a wider range of IoT devices, which includes PETs protocols. For this, one has to consider whether client applications can be made lightweight to ensure they can be run on resource constrained small devices. Such devices are limited in the types of computational operations they can perform, memory size, as well as by the battery life. We consider devices at least capable of performing symmetric encryption. Another important consideration is clients with an unstable connection, which makes them frequently join and leave the network. Many such clients are mobile devices such as mobile phones or wearables that cannot always stay in areas with good network coverage. Some of such clients are lightweight devices with very restricted battery power, or ones that are powered externally. These clients are forced to regularly disconnect.

In this thesis we investigate technological means for ensuring the privacy of communication on the Internet and in the Internet of Things. In particular, we focus on PETs that prevent disclosure of identifying attributes at both the communication and transaction levels. We examine several existing tools from the perspective of adapting them to the Internet of Things as it exists today, with the Internet being its primary network. When examining existing protocols or designing new ones we investigate primarily how the clients can be made lightweight and if the protocols can be adopted for mobile devices with unstable connections.

1.1 The scope of the thesis

The main research question of this thesis is:

> *How can PETs be implemented and adopted in IoT as it exists today in order to ensure privacy of communication by preventing identifying attributes from being revealed?*

The intention is to look at the problem from different angles in order to examine its complexity. The definition of privacy can be found in Section 2.1.

In pursuing our goal we consider two different communication networks that are common to the modern IoT. The first group form *Local Area Networks* (LAN) of nearby situated devices, often performed via ZigBee or near-field communication. This local communication happens when people pay with contactless debit cards, check in with their transport cards, or exchange information between two smartphones positioned next to each other. The second form *Wide Area Networks* (WAN) with devices located at large distances between each other, in particular the Internet. We consider several PET protocols that are applicable for use in these types of network.

We also focus on how the attributes of communicating parties are made available to various entities in a system. *Attributes* are commonly defined as qualities or parts of subjects or objects. Subjects are active entities such as senders and receivers, while objects are passive entities such as messages and keys. For example, an attribute can be the age of a person or his or her role as a student. In digital systems they serve as identifiers or role identifiers of subjects. This makes attributes essential for identification (establishing identity), filtering, authentication (proving identity or a role), authorization and many other essential functions in such systems.

Table 1.1 shows how the chapters of this thesis fit in described facets: attribute levels and types of network. Chapter 3 describes "ABC in IoT" intended for devices designed with services provided over the Internet. Chapter 4 describes "High-speed dating" that is designed for communication between RFID tags and a reader. Chapter 5 presents "Footprint scheduling" that is proposed for DC-net, an anonymous communication network well suited for both LAN and WAN communication. Chapter 6 presents Elligator, which can also be applied in both LAN and WAN communication. Finally, Chapter 7 introduces LightMix which is a variant of a mix-net, an anonymous communication network that is applicable to WAN communication.

Table 1.1: Representation of chapters of this thesis.

	LAN	WAN
Communication level	Chapter 4	Chapter 3
Transaction level	Chapter 6 Chapter 5	Chapter 6, 7 Chapter 5

Attributes of communicating parties. For the purposes of this thesis we differentiate between two levels of attributes, namely the *communication level* and the *transaction level*, which are covered in different chapters of this thesis. Attributes that belong to the first level are the ones that are inherent to all communicating parties. These attributes describe different roles that communication subjects can play and relationships between subjects and objects. These attributes include being:

- A sender or a recipient, which can be an end point or an intermediary;
- A communication partner of a particular entity;
- An originator of a particular protocol or a transaction.

The second level belongs to transactions, which we define as instances of communication performed for a particular purpose. Transaction level attributes are meaningful for this purpose. These attributes support decision making, for example whether an anonymous sender is authorized to request certain information or what kind of response he or she should get. Examples of such attributes are:

- Ownership of a device, an account, a subscription;
- Device manufacturer name;
- User age;
- Region where a user lives;

Without keeping communication level attributes private, the privacy protection of most Internet protocols and applications is questionable. For example, an anonymous payment system can be designed to allow participants to perform binding money transfers without revealing their identity. However, if their location is revealed, the anonymity measures of this payment system cannot protect users from identification.

Once communication level attributes are kept private, anonymization of the transaction level attributes helps sustain the privacy of individuals, while at the same time supporting functionality of many Internet applications and services.

1.2 This thesis at a glance

We focus on different levels of attributes in different parts of this thesis. In the part about communication level attributes, we examine systems designed to make such attributes private, namely anonymous communication systems (see Sections 2.2.1 and 2.2.2 for a general introduction to such systems). The three most popular systems in the literature are DC-net, mix-net and onion routing. They have all been extensively studied and are implemented in practical systems. We examine particular improvements that can be made to these systems. We investigate both DC-net and mix-net from the perspective of optimizing them for the needs of IoT. For onion routing we focus on meeting the current needs of Tor[1]. Tor is the most widely used implementation of onion routing in particular and anonymous communication systems in general.

Chapter 3.

Question answered	How can one prevent that transactions initiated by IoT devices are linked and tracked using revealed attributes?
Attributes level	Transaction level
Type of network	LAN
Type of device	Powerful, can afford asymmetric cryptography
Paper 	This chapter is based on the paper *"New Directions in IoT Privacy Using Attribute-based Authentication." [7] by Gergely Alpár, Lejla Batina, Lynn Batten, Veelasha Moonsamy, Antoine Guellier, Iynkaran Natgunanathan and me.*

[1] https://www.torproject.org

Contributions	I assisted in writing the paper. My contribution to the paper is the idea of the real-world problem which was described in the use-case section of the paper and served as a basis for the framework proposed in the paper. In collaboration with Gergely Alpár, I defined the precise notion of privacy in IoT that we use throughout the paper and analyzed threats to privacy following this notion. I actively participated in discussions that led to all other sections of the paper.

Chapter 4.

Question answered	How can lightweight RFID devices discover a match of attributes without revealing them to another RFID device or a reader? What can be done without asymmetric cryptography in such applications, and what can be done with asymmetric cryptography?
Attributes level	Transaction level
Type of network	LAN
Type of device	Lightweight, can afford no asymmetric cryptography, or a few public key encryptions
Paper 	This chapter is based on the paper *"High-speed dating: Privacy-preserving attribute matching for RFID." [25] by Lejla Batina, Jens Hermans, Jaap-Henk Hoepman and me.*
Contributions	I wrote most of the paper, developed the ideas for the protocols. The last protocol and security proofs of all protocols I performed in cooperation with my coauthors.

Chapter 5.

Question answered	Can anonymous scheduling in DC-net be adjusted to the frequent joining and leaving of participants that is natural for mobile devices while keeping communication overhead reasonable and providing the ability to scale well to the number of participants?
Attributes level	Communication level

Type of network	LAN and WAN. DC-net can be organized only by nodes
Type of device	Lightweight, can afford no asymmetric cryptography. Network connection is unstable, but has enough resources to actively send in the network.
Paper 	This chapter is based on the paper *"Footprint scheduling for Dining-Cryptographer networks." [202] by Moritz Neikes, Peter Schwabe and me.*
Contributions	I wrote about a half of the paper. I developed the ideas for the disruption detection protocol. My contributions include analysis of existing approaches to scheduling in DC-net, performance evaluation formulas. I actively participated in parameter tuning and analysis of footprint scheduling properties.

Chapter 6.

Question answered	Can one prevent an adversary from detecting the initiation of a key exchange protocol? In particular, can one make elliptic curve keys indistinguishable from uniform random strings?
Attributes level	Communication level
Type of network	LAN and WAN
Type of device	Powerful
Paper 	This chapter is based on the paper *"Elligator: Elliptic-curve points indistinguishable from uniform random strings." [48] by Daniel J. Bernstein, Mike Hamburg, Tanja Lange and me.*
Contributions	I assisted in writing the fist version of the paper. I participated in working out the proofs that demonstrate that the map Elligator 1 is a bijection from a large subset of $E(\mathbf{F}_q)$ to a large set S of b-bit strings.

Chapter 7.

Question answered	Can one adjust mix-nets for lightweight clients? In particular, how to minimize the amount of heavy cryptographic operations on clients?
Attributes level	Communication level
Type of network	WAN
Type of device	Clients can afford only limited number of public key operations, or it is done instead of them once during joining

Paper	This chapter is based on the paper *"LightMix: Mixing with Minimal Real-Time Asymmetric Cryptographic Operations." by David Chaum, Debajyoti Das, Farid Javani, Aniket Kate, Joeri de Ruiter and Alan T. Sherman and me.*. This paper is currently in submission.
Contributions	I wrote a substantial part of the paper. My contributions include performing a privacy and security evaluation in cooperation with Aniket Kate and Debajyoti Das. Protocol integrity protocols I developed in cooperation with David Chaum, Aniket Kate and Joeri de Ruiter. Comparison with other mix-net protocols I did in cooperation with Joeri de Ruiter. The main inventor of LightMix protocol is David Chaum.

Chapter 2

Privacy and privacy-enhancing technologies (PETs)

This chapter introduces preliminary information on topics of privacy and PETs that are used throughout this thesis. All other preliminary topics are introduced in the chapter they are directly related to.

2.1 Privacy

Definitions. Privacy is studied in depth in multiple disciplines, including law, philosophy, policy management, and computer science. All these disciplines shape different views on privacy, but one fact is commonly agreed on: privacy is a concept that is hard to define.

There are several famous definitions of privacy that are often used in the literature. We will introduce two that are most relevant to this thesis. One of them is quite commonly referred to in computer science. It was given by Westin [322] in 1970, who defines privacy as *"the claim of individuals, groups, or institutions to determine for themselves when, how, and to what extent information about them is communicated to others."* The popularity of this definition in computer science can be explained by the fact that it directly translates to the goals which most *Privacy Enhancing Technologies (PETs)* are created to achieve. More elaboration on PETs is given in Section 2.2.

Many PETs are designed to prevent disclosure and minimize the amount of information that can be revealed to or collected by third parties. When properly used,

such technology often provides to a user some degree of *control* that allows to define how his or her identity is represented in a digital system. This is achieved by defining circumstances under which some attributes of this person can be revealed, and which attributes are to be kept *confidential*. PETs provide such control even when facing a party that does not intend to respect user's privacy, or to comply with the rules of a given system or a protocol. We refer further in this thesis to such parties as *adversaries* or *attackers* interchangeably.

Another popular definition of privacy is given by Agre and Rottenberg [3]. Agre and Rottenberg defined privacy as *"the freedom from unreasonable constraints on the construction of one's identity."* Hilderbrandt argues in her work [168], that this definition includes an important aspect of privacy, namely the link between freedom from unwanted access to freedom to construct identity of an individual.

According to Hilderbrandt, a static conception of identity is useful for practical purposes of identification. It is built on defining the identity of a person by a set of attributes. Another approach to understanding identity is recognizing its developing nature that requires both active interaction with the outside world, and the ability to rebuild and reevaluate without intrusion. Hilderbrandt is referring to this unconstrained potential of changing as the indeterminacy of an identity. She sums up as follows: "Privacy empowers the human person of flesh and bones to rebuild its identity, by protecting its indeterminacy".

If one adopts such a definition in technology protecting privacy, spectrum of aims for PETs needs to be expanded to also capture the second privacy aspect, namely the freedom to construct identity. One of the severe ways this freedom is currently restricted is manipulation of available information. It includes filter bubbles, which are created as a result of algorithms producing personalized content, effectively confirming users' past experiences and viewpoints [249]. It also includes restriction to information access (censoring), actively performed by many companies and governments [238, 177].

Metrics. The privacy of a given system can be measured in two ways. One can either define it as a binary or as an ordinal variable. The first approach considers only existing or absent privacy. Such an approach is often taken within the cryptographic community , since it stems from the common cryptographic approach to defining security. In this approach one specifies a goal of a cryptographic tool. One also has to define abilities and goals of an adversary, the overall system and assumptions made. Having all that defined, one proves that the adversary is incapable of reaching his goal under given circumstances. Thus, if an adversary is capable of achieving

his goal, the cryptographic tool is considered to fail. Translated to privacy protecting measures, this means that if an attacker is capable of obtaining any information defined as sensitive, the protective measure is considered to have failed. Even release of partial information is often viewed as harmful, since "For most applications it is unreasonable to assume that the attacker forgets something. Thus, normally the knowledge of the attacker only increases." [260]

The second approach aims to distinguish and measure different levels of privacy. This approach makes it possible to decide if a given user or an overall system has more privacy than another. Quantification of privacy degree is common for data scientists [119, 299]. In anonymous communication networks such metrics also exist. Simple metrics consider to what degree an attacker can distinguish an IoI, for example, the given sender from the rest of possible senders [270, 260]. Another class of metrics uses entropy to estimate anonymity from the point of view of a single communication interaction [285, 109]. To estimate the global anonymity of a system, some studies [121, 147, 24] count the number of sensitive associations (e.g. between senders and their messages) an attacker has uncovered with high probability. They then ask: how does this number relate to the number of all possible associations in a system?

In order to approach viewing privacy as an ordinal variable, one has to make clear assumptions about what information is available to an attacker both inside and outside of the system. The measurement is correct only until such assumptions hold. Usually it is very hard, if possible at all, to make such assumptions in great detail about complex systems. It is even harder to verify the validity of such assumptions. Therefore in this thesis we use only the binary metric.

Terminology. As discussed earlier, privacy is an umbrella term that is covered in many disciplines. Implementation of a technical system requires terminology that makes it possible to differentiate fine-grained properties. For example, terminology devised by Vaudenay differentiates the privacy properties of a system based on the type of attacker this system can resist. This terminology is described in Section 4.3.1. We will use the terminology proposed by Andreas Pfitzmann and Marit Hansen [260] that is commonly accepted in PETs research community terminology. It is used in order to describe properties of protocols, as well as serves as a basis for specifying a goal of a cryptographic protocol. Terminology proposed by Andreas Pfitzmann and Marit Hansen considers the properties of a system consisting of users or subjects, and an attacker who can be one of the users or an external entity observing and/or modifying information flows.

Definition 1. *A subject or an object is **anonymous** if an attacker is not able to identify him or it from a set of subjects or objects with similar properties.*

Definition 2. *An **anonymity set** is the set of subjects or objects with similar properties, in which individual entities are non-identifiable by an attacker.*

In an anonymous communication system, one can have a separate sender anonymity set and recipient anonymity set. The sender anonymity set is the set of all senders who might have sent a message at a given time from the perspective of an attacker. Sender anonymity does not necessarily relates to senders of an application-level message; this can be an initiation of any network connection. Similarly, the recipient anonymity set is the set of all senders who might have received a message at a given time from perspective of an attacker. On the Internet it usually means that a user or a server can be contacted by those who do not know their network address, and an observer is unable to say which messages they received. Alternatively, this can be achieved if nobody knows what information a particular receiver is interested in. For example, this can be achieved when messages are broadcasted.

Definition 3. *An **Item of Interest (IoI)** for an attacker is an object or a subject which an attacker targets.*

This can be: the sender of a particular message, the receiver of a message, message content, the originator of a transaction, all transaction with the same originator, etc. Normally, a rule of thumb applies when one tries to improve the anonymity in a system. It states that the larger the anonymity set and the more even the distribution of probabilities of being an IoI (e.g., of being a sender) within it, the better the anonymity provided.

Definition 4. *Subjects and/or objects have the property **unlinkability** if an attacker is not able to identify a relationship between them.*

There are multiple examples of relationships that are often kept unlinkable in PETs. These include the relationship between a sender and a recipient, a sender and a message, a recipient and a message, two transactions, and a transaction and its originator. Often, one uses the term *untraceability* in place of unlinkability (of an entity and its location, or an entity and its activity). If an entity is anonymous, there is unlinkability between this entity and all other IoIs related to it. Sender anonymity implies, that any message sent in a system is unlinkable to this particular sender. At the same time, recipients of all these messages can be known.

Definition 5. ***Relationship anonymity*** *is achieved when there is unlinkability between both sender and recipient.*

If both sender and recipient know each other, but anonymity of both of them in relation to all other parties is maintained, this is called relationship anonymity. Note, this property is weaker than sender or recipient anonymity, because communication partners are not necessarily anonymous to each other.

Definition 6. ***Undetectability*** *is achieved when an attacker cannot detect if an IoI is present in a system.*

Definition 7. *Subjects and/or objects are* ***unobservable*** *if any subject:*

- *not involved in an IoI is not able to differentiate if an IoI exists or not (is undetectable);*
- *involved in an IoI is anonymous for other subjects involved in this IoI.*

Unobservability is the strongest property among all those described. Sender unobservability implies that a sender is anonymous for all users of the system, including an attacker and a recipient of his message. It also means that the recipient of the message is the only entity able to tell if some sender from the sender anonymity set has indeed sent a meaningful message at a given time.

2.2 Privacy-enhancing technologies

Privacy-Enhancing Technologies (PETs) are technological tools designed to support and provide privacy protection to the users of specific applications, while at the same time keeping the functionality, efficiency and usability of these applications at a desirable level. As mentioned in the previous section, many PETs achieve privacy by data minimization at the point in time when it is communicated or collected by third parties. For this PETs often rely on using cryptographic primitives and protocols.

2.2.1 A brief overview of research areas

There are a multitude of topics within the PETs research literature. In this section we give an (non-exhaustive) overview of some research areas related to PETs based on cryptographic techniques. PETs is a diverse area, thus many research topics

were not covered in this section, for e.g. steganography, location privacy, insurance pricing.

Anonymous communication. PETs created within this topic were developed to provide anonymity properties to the communicating parties (see Section 2.1). *Mix-nets* [80, 261, 258] usually provide sender anonymity, as well as sender-receiver unlinkability. *Onion routing* is a protocol based on the principle of mix-nets, adjusted to real-time communication with a reduction in the strength of protection [246, 151]. *Dining Cryptographers network (DC-net short)* [83] was also created by David Chaum. This PET provides both sender and recipient anonymity. These three anonymous communication systems are reviewed in greater detail in Section 2.2.2.

Anonymous credentials and e-cash. *Anonymous credentials* were invented by Chaum in 1983 in his work dedicated to anonymous payments [81, 82]. Later on they were further developed and renamed in the literature to *attribute-based credentials (ABC)* [73]. ABC makes it possible to reveal a selected number of attributes, while staying anonymous. An attribute can be any information about the user. For example, age, place of living, membership in a club, etc. The authenticity of such attributes is assured by its issuer. In the case of passport data, this can be a government, in the case of club membership, the club plays the role of an issuer. ABC also protects unlinkability between several transactions. A user can reveal the same set of attributes to the same entity several times, without anybody being able to identify that the transactions are initiated by same user. Anonymous credentials are reviewed in greater detail in Sections 3.3.3 and 3.5.1.

Anonymous credentials are used to implement *privacy-friendly digital cash systems* by making an attribute a coin, making its value the denomination of the coin, and then the payment is made by showing the credential. These systems do not only have to provide the unlinkability property between users and transactions, but they also prevent double spending of e-cash coins. This problem can be solved by having a third party verify and approve transactions, such systems are called online e-cash systems, for example [81]. Offline e-cash systems do not require any third party to perform a transaction, which makes the problem of double-spending harder to solve. To prevent double spending, offline e-cash systems make the user identifiable if the attribute is shown twice. This approach was proposed by Chaum, Fiat, and Naor [84]. Numerous further developments of e-cash have been described in the literature [61, 34, 72, 180, 283, 170]. E-cash systems dedicated to transportation can also be based on anonymous credentials [167, 280].

Not all e-cash systems are based on anonymous credential; a widely used called Bitcoin [233] is one of them. Bitcoin does not provide anonymity [54]. A number of papers focus on adding transaction unlinkability to it. ZeroCash [38] maintains the decentralized property of Bitcoin and the efficiency of ZeroCash is improved in [102, 143]. The drawback of these systems is that they modify the protocols used in Bitcoin, which can make adoption problematic [333]. Another approach is to involve techniques from anonymous communication systems. MixCoin [57], and CoinParty [333] utilize a mixing server, CoinShuffle [279] and SecureCoin [175] uses mix-net structure, while CoinJoin [223] uses Tor [113].

Private Information Retrieval (PIR). This PET can be viewed as a part of anonymous communication, because retrieval can be used to provide receiver anonymity. PIR makes it possible to retrieve data from a storage location without anyone knowing what was retrieved, including the storage itself. PIR requires the user to know the exact address of the data item he or she is interested in. An exception to this is the PIR based on keyword search [195].

PIR was first introduced in 1995 [86], and is called information-theoretic, which means that no server has any information about the data item. Instead, this data item is distributed over multiple servers and can be retrieved only if data from several such servers is collected. The security of this scheme relies on the assumption that these servers are not colluding. A number of schemes with information theoretic property have been reported in the literature [28, 27, 29, 149]. Single server solutions rely on cryptographically hard problems [204, 71, 225]. A hybrid approach improves on both types of PIR, the information theoretic one and the one that relies on cryptographically hard problems [107].

An *oblivious transfer* [62, 210, 17] can be considered to be a PIR scheme with stronger guarantees. Like a PIR, it makes it possible to receive information such that the user receives one out of several transferred data items without the server knowing which of the data items were received. At the same time the user does not know the other data items that he or she received. A single-database PIR with total communication that is less than the size of the data stored can be reduced into oblivious transfer [108].

Censorship resistance. *Censorship-resistant publishing systems* are designed so that they make it possible to anonymously publish and access sensitive documents without a risk that any of the documents will be blocked within such system. Examples of such systems are Free Haven [112], Freenet [87, 277], GNUnet [41, 203], and Anon-Pass [207]. The disadvantage of the censorship-resistant publishing systems

is that they reveal the whole list of available servers to any user, which makes it easy to automatically block access to all of these servers [201].

Communications censorship is provided by proxies and anonymous communication systems [80, 261, 258, 205, 246, 151] can be used to bypass simple blocking based on IP address of the destination. To avoid more advanced censoring techniques, other protection techniques need to be added. Two major trends exist, one of which protects the set of servers that forward user's traffic from being easily learned by the censor [201, 173, 189, 124, 55]. Another approach is to create shape traffic to look like another application, for example VoIP [193, 174, 212, 120, 313, 306].

Systems that are specifically developed for providing censorship resistance often include a covert channel by design and focus on plausible deniability for their users. Covert channels are communication methods that are built to let only communicating parties to recognize that communication is taking place. For example, Infranet [131] is a system of forwarding servers that receive requests shaped as HTTP traffic and send replies via convert channels in images. Telex embeds covert channel in TLS handshakes [326].

Privacy-Preserving Data Mining (PPDM) and Privacy-Preserving Data Publishing (PPDP).

These types of PETs focus on preventing leakage of sensitive information about data sources within a dataset. PPDM puts emphasis on a model with any data being queried, while PPDP is used for sanitizing data for release. However, this distinction is not strict, and both PPDM and PPDP are often used interchangeably in the literature [247]. An overview of this field can be found in [235, 220, 263, 215].

Statistical Disclosure Control (SDC) techniques are needed to prevent an attacker from learning from disclosed data more information about the data source, or completely identifying him or her. This is usually referred to as re-identification. The main goal of SDC is to minimize the risk of re-identification, without harming the data utility. To achieve this goal a number of papers suggest masking the original data. Alternatively one can produce synthetic data such that the statistical properties of the original data hold [224, 271].

Perturbative masking methods modify original data in order to achieve anonymity. Examples of such an approach include adding random noise to the data [118] and swapping attributes between data items [132, 289]. Non-perturbative masking avoids altering original data by partial suppression or reduction of details. Several techniques have been devised to reduce the uniqueness of the attributes of data

items, such as generalization and local suppression [299]. These techniques are used to achieve k-anonymity. A data set has *k-anonymity* when every attribute value is represented in at least k data items (e.g. rows in a database table) [299, 308]. Thus, a single data item is indistinguishable from any other $k-1$ data items. A number of studies have introduced improvements [116] to k-anonymity, including l-diversity [218, 192], t-closeness [211] and β-likeness [77]. Another notion is ϵ-*differential privacy* [119, 281], which works by estimating if adding even a single data item will have any significant influence on the result of a query on this data. ϵ-differential privacy is currently considered to offer a stronger privacy protection than k-anonymity and its improvements at a price of more limited data utility [282]. A comparison of two families of privacy notions can be found in [89].

Secure Multiparty Computation (MPC) ensures that several parties can perform a collaborative computation without revealing their individual inputs. Within PPDM MPC is used to perform joint data mining without disclosing the datasets of participants [215, 263]. A detailed study of the set of basic operations on data (e.g. sum, set union, etc.) that can be performed efficiently, as well as their application to PPDM is discussed in [88]. Homomorphic encryption is one of the popular ways to construct MPC. This type of encryption can be used to encrypt data such that it can be mined in its encrypted form. After decryption, the resulting data will be same as if the mining process was performed on the original data. Partially homomorphic encryption has a one of the basic operations such as addition and multiplication which it supports. Fully homomorphic encryption can sustain arbitrary operations [145]. Both types of homomorphic encryption are used for collaborative computations in PPDM [325, 159, 332, 95].

2.2.2 Anonymous communication systems

Distributing trust among several servers or users in an anonymous communication system is a default approach, making it possible to mitigate many insider, denial of service, and local traffic-analysis attacks [100]. A large number of state-of-the-art systems stem from either mix-nets, onion routing or DC-nets. Both mix-nets and DC-nets resist a global attacker, while onion routing relies on the restriction that an attacker cannot observe all the links in network. Mix-nets and onion routing require several central nodes that process messages encrypted in layers. In addition, mix-net nodes add random permutation of incoming messages and sometimes a delay before sending a message further. Table 2.1 presents an informal comparison of original versions of these systems [80, 151, 83]. The anonymity of DC-net is *uncondi-*

tional, because it doesn't rely on computationally hard mathematical problems, unlike mix-net or onion routing. However, it requires broadcasting to be reliable [316]. The computational load is light in DC-net, because it only requires xor operations. Both of the other systems use asymmetric cryptography. The network load is extremely heavy in DC-net because, in order to transmit a single meaningful message from a user, it requires all other participants to transmit meaningless messages of the same length. DC-net can have a central server that publishes anonymized messages, however, it can function as well without [258]. Mix-net or onion routing requires servers to forward messages of users. Data retention is the storage of network communication data and meta-data by communication providers. Data retention row presents an ability to store sensitive data that can be used for further deanonymization of users. No information stored on a DC-net server can be used to deanonymize it's users, but this is not the case for mix-net or onion routing. Both DC-net and mix-net can resist a global adversary who is capable of seeing all the links in the network.

Table 2.1: Comparison of basic DC-net, mix-net and onion routing systems.

PROPERTY	DC-NET	MIX-NET	ONION ROUTING
Anonymity	unconditional	computational	computational
Computation load	lightweight	moderate	moderate
Network load	extremely heavy	normal	normal
Network architecture	central or peer-to-peer	central servers	central servers
Data retention	not possible	possible	possible
Attacker	global	global	not global

Mix Networks

In 1981, Chaum [80] introduced the concept of mix-nets (often referred to as *decryption mix-nets*) and gave basic cryptographic protocols whereby messages from a set of users are relayed by a sequence of trusted intermediaries, called *mixnodes* or *mixes*. Before being sent to the first mixnode, messages are padded and encrypted in several layers. Each layer is encrypted using a public key of the mixnode through which the message has to travel next. A mixnode is simply a message relay (or proxy) that accepts a batch of encrypted messages of equal length, decrypts and

randomly permutes them, and sends them on their way forward. Figure 2.1 depicts the mix-net with three mix-nodes, four senders and receivers.

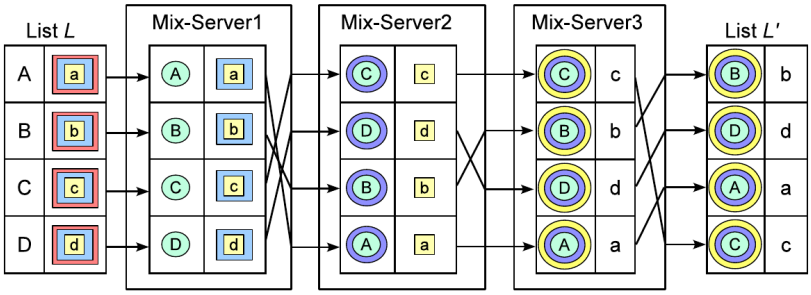

Figure 2.1: Mix-net with tree mixnodes. Figure is taken from [331].

Both decryption and random permutation are used to make sure that it is impossible to establish a link between incoming messages of a mixnode and outgoing, without knowing the secret key and the permutation. Decryption of messages that have equal length ensures that no recognizable pattern of bits is left when messages leave mixnode. Random permutation makes sure no link can be established based on the order of messages.

Due to incoming messages being unlinkable to the outgoing ones, senders also are unlinkable to messages output of mix-net. Mix-net maintains this unlinkability against an external attacker if there is at least one mixnode. If an attacker is internal, there needs to be at minimum one server that is not under control of an attacker.

The sender in a decryption mix-net must perform a number of public-key operations equal to the number of mixnodes. The length of the resulting encrypted message is proportional to this number, and the length of the plaintext message is thus restricted for performance reasons.

Hybrid mix-nets allow plaintext messages to have arbitrary length, by combining asymmetric and symmetric cryptography. First proposed in 1985 by Pfitzmann and Waidner [258, 256], hybrid mix-nets share a session key in the header of message, and then use a stream cipher to encrypt the rest of the message. Various proposals based on block ciphers followed [242, 183]. The recent system called Riffle [205] provides both sender and receiver anonymity by using verifiable shuffles and private information retrieval. Verifiable shuffles [236] allow to perform secretly shuffle a set of messages in such a way, that correctness of this operation can be proven. Periodically, a client and all servers perform verifiable shuffles to exchange session

keys, which are then used for several rounds in a similar manner to that performed by other hybrid networks.

In 1994, Park et al. [250] introduced *re-encryption mix-nets*. Taking advantage of homomorphic properties of ElGamal encryption, each mixnode re-encrypts the incoming message instead of decrypting it as in the original mix-net. Doing so reduces and fixes the size of the ciphertext message traveling through the mix-net. *Universal re-encryption mix-nets* [155] do not require mixnodes to know public keys for re-encryption. Because the sender encrypts each message using the public key of the receiver, only the receiver is able to read the plaintext. Consequently, the recipient can trace messages addressed to them directly from the point when they were sent by senders. Thus, unlike other mix-nets, universal re-encryption mix-nets provide sender anonymity only against external observers, and not against message recipients.

Precomputation mix-nets introduce a precomputation phase to decrease latency during message delivery. Jakobsson [182] introduced precompution to reduce the cost of node operations in re-encryption mix-nets, though client costs remain the same. Adida and Wikström [1] considered an offline/online approach to mixing. Their protocol still requires several public-key operations in the online phase, and senders have to perform public-key operations.

Onion Routing

Higher latency of traditional mix-nets can be unsatisfactory for several communication scenarios, such as web search or instant messaging. Over the past several years, a significant number of *low-latency* anonymity networks have been proposed [245, 74, 85, 191, 190, 150, 23, 22], and some have been extensively employed in practice [304, 113].

Common to many of them is *onion routing* [246, 151], a technique whereby a message is wrapped in multiple layers of encryption, forming an *onion*. A common realization of an onion-routing system is to arrange a collection of onion routers (abbreviated ORs, also called hops or nodes) that relay traffic for users of the system. Users then randomly choose a path with few edges through the network of ORs and construct a circuit—a sequence of nodes that will route traffic. After the OR circuit is constructed, each of the nodes in the circuit shares a symmetric key with the anonymous user, which key is used to encrypt the layers of future onions. Upon receiving an onion, each node decrypts one of the layers, and forwards the

message to the next node. Onion routing as it typically exists can be seen as a form of three-node mixing.

Low-latency anonymous communication networks based on onion routing [232, 296, 127, 184], such as ToR [304], are susceptible to a variety of traffic-analysis attacks performed by a global attacker [232, 268], and by the one who can observe only traffic of users [166, 319]. By contrast, the mix-net methodology ensures that the anonymity set of a user remains the same through the communication route and makes mix-net resistant to these network-level attacks.

Dining cryptographers

Dining cryptographer networks is another anonymous communication system proposed by David Chaum [83]. Multiple modifications have been proposed since then. Herbivore [148] is an implementation of DC-net that organizes the network into several groups with small anonymity sets. Golle and Juels [156] propose two new versions of DC-net that substitute xor with public-key encryption, which in turn makes it possible to verify the correctness of computations with zero-knowledge proofs. Fully verifiable DC-nets are proposed by Franck [138] and in Verdict [94]. Dissent [93] provides provable anonymity. It has an extension aimed at improving performance at the cost of having to rely on one server to be honest [324] (the same model is used in Verdict).

To explain how DC-net works, consider an example with three participants exchanging 6-bit messages. Figure 2.2 depicts this example. Each participant has a shared symmetric key with each other participant. Assume that participant A wants to send a message. To do so, she xors her message with all the keys she shares with the other participants. The result is an *output* that A sends out to every participant of the DC-net. Note, each key has the same size as the message, and is used only once, thus the output is essentially a one-time-pad encryption.

The other participants perform the same procedure, but use a zero message instead of a meaningful one. All outputs from all participants are xored together to reveal the *global sum*. The global sum results in the meaningful message from the participant A, because the keys cancel out (since each symmetric key is used twice). This process is referred to as *superposed sending* [258].

Performing a xor of all participant's outputs and broadcasting the resulting message completes a single *round* of the DC-net, during which one participant can transfer (broadcast) one message; in the next round another participant transmits a message until all participants are done transmitting. This explains the performance

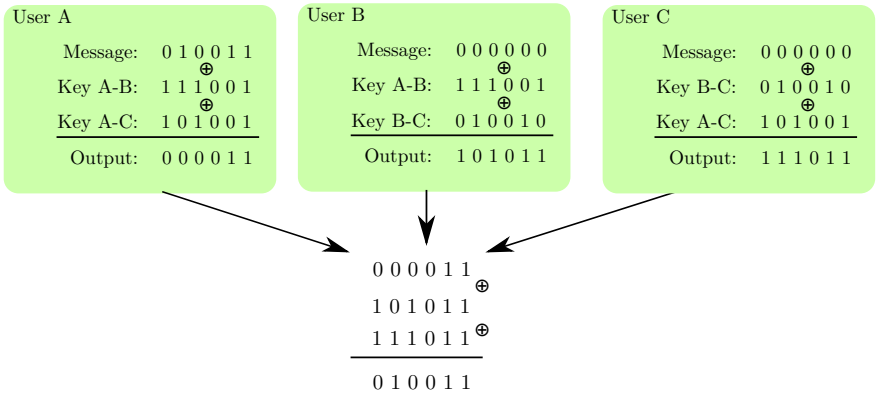

Figure 2.2: DC-net with tree participants and 6-bit messages.

problems of DC-net – in order to let a single participant transmit his or her message, each other participant has to transmit meaningless output of the same size.

Collisions and scheduling. If two participants send a message in the same round, their messages collide and become unusable. This can happen accidentally or also intentionally by a malicious participant who is *disrupting* communication. This raises an important question: How does each participant know when it is his or her turn to send? Note that many standard slot-reservation protocols as used, for example, in dynamic time-division multiple-access (TDMA) networks are not applicable, because they would compromise anonymity. The task of *slot reservation* in DC-net is to agree on a sending schedule in such a way that each participant knows when to send, but does not learn who is sending in the other slots.

Generally there are three different approaches to solving this problem. The first approach comprises so-called *reservation-map* methods. These methods employ a *scheduling vector* consisting of *slots*; each slot represents a future round with a corresponding index number. To reserve a round, a participant marks the corresponding slot in the scheduling vector as occupied. The second approach is to use *collision-resolution algorithms*. The third approach is to use secure multiparty computation to obtain a secret permutation that assigns rounds to participants. Section 5.2 provides a detailed overview of these scheduling methods.

Sender anonymity. Superposed sending ensures sender anonymity, which is proven to be information-theoretic by Chaum [83]. This holds with at least two honest participants sharing a key. Andreas Pfitzmann extended the protocol from $\mathbf{GF}(\mathbf{2})$ to

addition in a finite abelian group $\mathbf{G}$ and proves that it also provides information-theoretic anonymity [259].

Superposed sending in $\mathbf{G}$ is performed in the following way. The set of N participants of the DC-net share secret keys with each other. These keys k_{ij} for $i, j \in 1, 2, ..., N$ are chosen uniformly at random from $\mathbf{G}$. Since the keys are shared between participants P_i and P_j, k_{ij} and k_{ji} are equal.

Assume that participant A plans to transmit a message $m_1, m_2, ..., m_B$ with each character $m_t \in \mathbf{G}$ and $t \in \{1, ..., B\}$. To produce an output, the participant A applies the following equation to all B characters of his message to obtain corresponding characters of the output:

$$O_t = m_t \oplus \sum_{\{P_i, P_j\}} sign(i-j)k_{ij}\,, \tag{2.1}$$

where $sign(i-j)k_{ij}$ is an inverse of k_{ij} if $i < j$; otherwise it is equal to k_{ij}. The concatenation of resulting characters is the output of participant A. Each character S_t of the global sum is calculated from all N corresponding characters of local sums O_t:

$$S_t = \sum_{j=1}^{N} O_j\,. \tag{2.2}$$

Each character of the global sum is equal to the sum of all corresponding message characters m_t. The keys have no influence on the final result since they were subtracted after being added. Given that only participant A used non-identity characters in his or her message, the global sum will be equal to that message.

Receiver anonymity. The message resulted from superposed sending is broadcast to all participants, effectively keeping the receiver anonymous. Instead of broadcasting, other techniques that provide receiver anonymity can be used, in particular Private Information Retrieval (PIR) [86]. In the rest of this chapter we consider DC-net with broadcasting. This broadcast network has to be *reliable*, that is all messages sent from honest participants arrive at all other participants unmodified; otherwise the active attack on DC-net described in [316] becomes possible. The attack works as follows. An adversary partitions the anonymity set by sending to one part the original message m that resulted in the round, and to the other side some other message m'. In the coming rounds the adversary observes an answer to either m or m', for example he knows that the message m' dictates a certain answer based on the protocol of communication. Depending on his observations, he can identify

the set to which one of the communication parters belongs. Given that the participants communicate over a long period of time, this attack can be repeated further, leading to complete deanonymization.

Waidner [316] proposed using fail-stop broadcast technique to prevent such attacks. It guarantees that either all participants receive the same broadcast in the given round, or all the consecutive rounds become jammed. This is achieved by making the keys of any round depend on the message broadcast in the previous rounds. Thus, if at least two participants sharing a key receive different broadcast, their keys will desynchronize and will no longer cancel each other out in the global sum. As a result, the global sum will contain rubbish instead of a meaningful message.

There are a number of challenges that arise when fail-stop broadcasting is used. Firstly, it is hard to distinguish whether a round was disrupted, or was jammed due to the fail-stop reaction to an inconsistent broadcast. Without special measures, it is impossible to identify a disruptor without a high chance of false accusations of honest participants. Even if all the keys for a given round are open and verified, honest participants whose keys are desynchronized due to fail-stop broadcast will seem to be adversaries. Secondly, fail-stop broadcast enforces a sequential run of the protocol, preventing precomputation. Without fail-stop broadcast one can precompute outputs or commitments to outputs that prevent an attacker adjusting his output according to the outputs of other participants he observes [80].

Chapter 3

New directions in IoT privacy using attribute-based authentication

3.1 Internet of things

The term Internet of Things (IoT) became widely used after it was presented by Kevin Ashton in 1999 [18]. The concept itself appeared earlier in an article by Mark Weiser [321] under the name ubiquitous computing. Other frequently used names are pervasive computing [240] and ambient intelligence [162].

In addition to having multiple names, IoT is a fuzzy concept that has a multitude of definitions. The IEEE supported an attempt to analyze existing definitions and unify them. As a result, the following definition emerged: *"Internet of Things envisions a self-configuring, adaptive, complex network that interconnects 'things' to the Internet through the use of standard communication protocols. The interconnected things have physical or virtual representation in the digital world, sensing/actuation capability, a programmability feature and are uniquely identifiable. The representation contains information including the thing's identity, status, location or any other business, social or privately relevant information. The things offer services, with or without human intervention, through the exploitation of unique identification, data capture and communication, and actuation capability. The service is exploited through the use of intelligent*

interfaces and is made available anywhere, anytime, and for anything taking security into consideration." [274].

For a long time IoT was only a vision of a future technology. This is not the case nowadays. "It's not science fiction anymore. It's science fact", states Boo-Keun Yoon, president and CEO of Samsung Electronics in 2015 [194]. Estimates made in 2016 show there are 6.4 billion connected things, only slightly less than the total number of people on earth [12]. By another estimate a digital consumer on average owns 3.64 connected devices [70].

IoT, as it is described by futurists, is attractive. It promises smart resource management though intelligent monitoring, control and notification. For example, smart energy or water meters in households can automatically track energy consumption and switch devices on or off as needed. Systems on fruit farms can notify farmers in time about important conditions such as the emergence of fruit flies that are capable of destroying a large portion of the harvest. IoT can help optimize the production, storage and delivery of products through automation of processes such as machinery maintenance. IoT can also help optimize daily behavior of people with smart metering, notifications and suggestions on devices such as smart watches.

There are numerous possible IoT applications. There are also a multitude of existing IoT products. They differ from predicted IoT products in that they are usually narrowly focused in terms of applications and are not yet interoperating. For example, home automation systems such as Belkin WeMo [30] and Logitech Pop [216] make it possible to remotely control smart wall switches and plugs, LED light bulbs, motion sensors and lighting devices. Nest [237] produces smart connected thermostats, smoke detectors, cameras and other home security systems. Smart watches such as Pebble [253] make it possible to track daily activities, calorie intake, sleep patterns, fitness exercises and much more. Despite existing IoT products being profitable, the main interest in IoT for businesses lies in existing business models related to Big Data. IoT is primarily valued as a source of a large amount of new information which companies can collect about their customers [264].

The goal of collecting large amounts of data is supported by the centralized architecture present in almost all current IoT products. The collected data cannot be stored on the smallest devices comprising IoT, thus IoT makes extensive use of backend systems. Lightweight devices connect through a hub to a central server or a cloud, which holds the data and a large amount of programmed logic. Some sources even state that cloud technology is what enables IoT [302]. Propos-

als for clouds aimed at IoT are plentiful: Oracle cloud [91], Google cloud [90], IBM cloud [179], to name a few. Every single one of them advertises their data analysis capabilities on Big Data.

IoT is a complex heterogeneous system, comprised of a variety of different devices and communication protocols, as well as different types of networks. Devices in IoT are divided into three groups: sensors, actuators and smart devices. Sensors are made to monitor and report certain parameters in the environment. For example temperature, motion or light. Actuators are devices that can interact with physical objects in the environment. Smart devices control both sensors and actuators in order to provide services to users.

Devices also differ in terms of how computationally lightweight they are. On one side of the spectrum are medical nanowire sensors [251] and $0.04\ mm^2$ RFID chips [172] (see Section 4.2.1 for an introduction to RFID), and on the other side there are computationally capable devices such as mobile phones. The smallest devices often do not have their own battery and need to be powered externally, while others carry their own battery and are limited by its lifetime.

Communication protocols also depend on the type of devices used and include near-field communication used by RFID tags and readers, Bluetooth, ZigBee, Z-Wave, 6LowPAN, Wifi and many others. The same holds for types of network. A Body Area Network (BAN) is a network consisting of wearable and medical devices positioned inside or on the surface of the body. A Local Area Network (LAN) consists of all devices located in close proximity, typically a house. Smart home appliances can compose a single LAN network. A Wide Area Network (WAN) connects devices located at large distances. The Internet is a WAN that can be considered as part of IoT since backends, clouds, and small devices are connected to it. Thus, when investigating IoT, one has to take into consideration both lightweight and computationally capable devices. Likewise, one has to consider applications and protocols designed for local area communication, as well as the those designed for WAN, including existing Internet applications and protocols.

The Internet plays – and will likely keep on playing – a major role in IoT. Industry mostly defines IoT from the angle of connecting things to the Internet [274, p. 40]. In particular, things often connect via the Internet to backend systems such as central servers and clouds. Backend systems allow constrained lightweight devices to store data and outsource some of the logic. Additionally, they ease the collection and analysis of Big Data generated by IoT products.

3.2 Authentication in the IoT

The Internet of Things is becoming a part of our daily lives. Gartner predicts [273] that within five years there will be more than 25 billion devices that take part in the IoT. Because the IoT delivers services in a fast and convenient manner to users, it is viewed as a major business opportunity [265].

Most IoT technologies rely on a centralized architecture, in which sensors collect data from the world, communicate to computers, which in turn send the data to a central service. This architecture enables efficient operation and full control over the data processing and dissemination, which makes it an appealing approach for businesses. However, privacy concerns arise since users cannot control the information that is collected about them. Furthermore, organizations may be subject to legal problems in terms of privacy regulation. In our opinion and that of the authors of [265], in order for an IoT architecture to be sustainable, it must be decentralized and user-centric.

When devices communicate in the IoT, they often need to authenticate each other, so that they know that they talk to the intended counterpart. Typically, authentication (proving that the communication partner is who he or she claims to be) involves identification (claiming of an identity) by means of a unique number or name and a token (*e.g.,* a password or a cryptographic proof) that proves the validity of the identifier. On the one hand, identifiers make authentication easy, provided that the verifier has access to a database of these identifiers. On the other hand, identifiers make all the transactions carried out by a particular device linkable. Moreover, in many cases devices are associated with an individual; so, indirectly people become traceable as well. Based on the increasing amount of user data, organizations, including advertisers, can build profiles about people. This is obviously harmful to people's privacy, and in many countries, the use of data for tracing and profiling is contrary to legislation governing the so-called secondary use of data [292].

With regard to user control, data collection can be effected in two ways in the IoT [278]. First, there are devices which the user can control how information is collected. Examples include wearables and smart home units. Second, there are devices for which the user has no control over the data collected; these include sensors and surveillance cameras in public places for instance. This technical distinction has to be taken into account when defining privacy and when designing new IoT technologies.

IoT applications often rely on data analysis tools which can recognize patterns and extract further useful information from already collected data. For instance, analyzing energy usage of a neighborhood can help distribute power efficiently within a city. Such analysis tools are also suitable to discover personal information that users may want to keep private. An important way to decrease the privacy risk caused by data analysis is to reduce the amount of data collected, as proposed by the European Parliament and Council in their *data minimisation* principle [126].

Research in identity management [11, 267] shows that attribute-based authentication can realize data minimization with possible user control. Attributes, characteristics or qualifications of entities, are embedded in a cryptographic container which can be used for authentication purposes. Since attributes such as the brand of a device or the nationality of a person, can be anonymous, authentication can also be anonymous. In fact, authentication should, and can, be realized by using the minimum amount of information required to successfully complete a transaction.

3.2.1 Our contribution

We recognize the importance of the right of the individual to have control of their own data and not to have their transactions linked and tracked within the IoT. On the other hand, the threat of mass surveillance and of linking users and their devices to transactions is very real within the IoT.

In this research, we present a framework for providing the user with the ability to avoid linkability and at the same time, maintain control of their data. Our solution is based on AB authentication instead of identification to guarantee authenticity in the communication among devices in the IoT. We demonstrate the feasibility of our framework by introducing it into a common use case, in a home environment.

We also argue that AB credentials should be considered in every scheme where authentication is needed, as it will provide the user with increased privacy and control with respect to their personal information.

3.3 Related work

In this section, we present some of the recent work related to authentication and privacy in IoT, and the uses of attributes.

3.3.1 Authentication and privacy in IoT

Position papers on IoT such as [301] and [146] often include discussion on authentication and privacy. In fact, many such papers point out the authentication and privacy problems arising from ubiquitous presence of sensors and devices, and the subsequent analysis of massive amounts of collected data.

As observed by the authors in [4, 222, 2, 334], one of the capabilities of IoT is to allow sensors to collect information from their surroundings, record and process it. With an on-going surge in the number of inter-connected devices [273], the tendency of collecting more information than required for the provision of a service is on the rise.

Conversely, one of the business drivers of the IoT is expressed as "improved customer retention and more targeted selling" [222]. This makes way for a strong tension between the need for massive data collection, processing and communication required for the services of IoT on the one hand, and privacy protection of individuals on the other [222, 200]. Moreover, multiple studies have shown that collection of even seemingly innocent data can lead, with very high probability, to the identification of individuals [297, 298, 234].

3.3.2 Identifiers in IoT

While interacting with an IoT architecture, users perform identity management [10] implicitly. They produce personal information and leave traces mostly bound to their identity. For instance, adjusting your house's temperature using a mobile phone requires that the system knows that the instruction came from a legitimate party. Typically, a system stores a lot of information about its users. The set of all this data with respect to a particular individual is her *identity* in this system. In most cases at least one of these pieces of information acts as an *identifier*, that is, a direct link between the individual and the system.

The most common way to authenticate individuals and devices and authorize them for services in today's Internet is by using *Federated Identity Management* [79]. These solutions involve an *Identity Provider* and one or more *Service Providers*. When a user needs to authenticate to a service, the identity provider intervenes to *assert* the user's identity to the service provider. This allows for flexible authentication and the decorrelation of identities and services. However, the Identity Provider, being involved in all transactions, can trace users connecting to services.

3.3.3 Use of attributes

Authentication protocols often reveal more about the user than necessary. Indeed, in many cases, only an assertion on the user's *attributes* is really needed. For instance the only information that may matter is that the user belongs to a registered service provider, or has a sufficient clearance level. In this sense, identity-based authentication does not comply with the *data minimization* principle as prescribed by *Directive 2006/24/EC of the European Parliament*. Existing identity-based authentications solutions however already admit that attributes are what really matters, since after obtaining the user's identity, the SAML standard [244] (in particular) allows the service provider to request specific attributes (or *claims*) about the user [79].

The ABC4Trust project [267] demonstrated that the *attribute-based credentials* (ABC) technology is an adaptable means of realizing both flexible and privacy-preserving authentication. Indeed, the identity provider needs only to be online when the credential is delivered to the user; this prevents the identity provider from profiling users based on their authentication patterns, and from impersonating them. In addition, the user has control over which attributes are used thanks to the *selective disclosure* mechanism of the ABC constructions.

The main cryptographic schemes putting ABCs in practice are *U-Prove* [248] and *Idemix* [75]. They have been implemented on smart cards [231, 314] resulting in a U-Prove authentication processed under 1 second, while that in Idemix is between 1 and 1.5 seconds. These technologies have also been placed into light-weight infrastructures in [8, 9, 267]. However, none of the previous research projects proposes to adopt ABC technologies within the IoT.

3.4 New directions in IoT privacy

3.4.1 Privacy threats

The usual approach in cryptography and security areas is to define an adversary by his goals and capabilities. However, in data collection the initial goals may be set with the best intentions of a data collector, but possession of the accumulated data can be transferred rightfully (or not) to a data processor that does not share good intentions of the original one. Hence, the adversarial goals are rendered irrelevant during data collection phase but can be of impact during data processing and data dissemination phases.

Data over-collection [213] and the *always-identify* paradigms described in the Introduction are other existing privacy threats that become increasingly harder in IoT. Coupled with *linkable transactions*, that is transactions the origins of which are known to be the same [260], these threats enable extensive user profiling. For instance, service providers can create a *fingerprint* of an individual based on the types of devices he utilises, as presented by van Deursen [310] in the case of RFID tags. This may give a lot of information about the device owner; moreover, this fingerprint acts as a new identifier, making it impossible for the user to remain anonymous later.

3.4.2 Defining privacy in IoT

We argue that the *lack of control* is one of the central problems in IoT in terms of privacy. Individuals become a part of a pervasive computing system, and by that, they generate a lot of personal information while having only limited oversight and control over data collection and processing. We adapt the definitions by Alan Westin [322] and Ziegeldorf *et al.* [334] as the former does not address the problem of data collection and data processing, and the latter places too much responsibility on the data subject.

Definition 8. *Privacy is the right of individuals to determine for themselves when, how and to what extent information about them is collected, processed and communicated; that includes individuals having*

- *the right to determine these aspects within their area of control explicitly;*
- *trust that the right above is respected when control is not possible.*

Following taxonomy of privacy by Solove [292], privacy threats are raised during data collection, data processing and data dissemination activities and intrusions. Because malicious intrusions form a whole body of work separate from privacy, we do not discuss them here. Specific to IoT, data collection is happening on a massive scale, much of it is collected by sensors without any active initiation from the user, thus often leaving users unaware of this process. The large amount of data and the high number of potential data sources increases severity of privacy threats at data processing and dissemination activities compared to the today's situation in the Internet.

In the present, control over data processing and dissemination activities is largely in the hands of lawyers and policy makers. From the users' perspective, technologies providing them meaningful control are virtually non-existent. Although

some Privacy-Enhancing Technologies (PETs) can provide a solution to this problem, their use is left at the discretion of companies and agencies, who only employ them when regulation mandates so.

The control over data collection, on the contrary, is partially in the hands of users. Indeed, although control is impossible in the presence of sensors (such as surveillance cameras) that collect user data in a passive manner, possibility to control appears when the user or any of the devices acting on his behalf (and that are under his control) are actively engaged in a communication.

Altogether this makes privacy-friendly data collection an important stepping stone towards achieving privacy in the IoT. First of all, this is the only area in which it is feasible to implement some level of technological control from the user's perspective. Secondly, the increased amounts of collected data leads to correspondingly increased levels of threats during data processing and dissemination. Thus, addressing this particular point – data collection – one can provide significant increase in the privacy protection level of an individual user.

3.5 Realising IoT using attributes

3.5.1 Attributes and attribute-based credentials

Conceptually, attributes are properties or qualifications of an entity. In practice, an attribute can be anything that can be described as a bit-string, such as a name, a date of birth or a cryptographic token. In this respect, attributes generalize the notions of identifiers and roles.

An ABC [248, 75] is a cryptographic container of attributes. Similar to a traditional X.509 certificate, it is issued by a trusted party, and is bound to a specific entity *via* this entity's secret key. That is, the issuer cryptographically signs a token consisting in the concatenation of the attributes and a *commitment* to the entity's secret key. For instance, a service provider may issue to each of its clients a subscription ABC containing a client number, a specified purchase level and the start date of the subscription. The issuer is trusted for verifying that the attributes in an ABC authentically belong to the entity. Having an ABC, an entity can show its attributes and *prove* that they are signed by the issuer. This is done using (non-interactive) zero-knowledge proofs to guarantee that no other information (*e.g.* some leakage about the secret key) is revealed.

In addition to the traditional *unforgeability* property, ABCs come with many privacy-enhancing mechanisms [5, Chapter 3]. First of all, because the proving of attributes is performed in zero-knowledge, the *verifier* does not learn the (commitment on) the secret key of the prover in the process. Even better, some ABCs scheme have the *multi-show unlinkability* property that prevents a verifier from linking two showings of the same ABC by the same entity (except if the content of attributes themselves leak information). Secondly, the *selective disclosure* functionality allows an entity to demonstrate only an arbitrary subset of the attributes contained in its ABC. Continuing the subscription ABC example from above, a client can choose to show only its purchase level, which may grant discounts and advantages, but not its client number.

3.5.2 Privacy using attribute-based authentication

Privacy threats can be thwarted thanks to the privacy-friendly properties of ABCs. First of all, the *always identify* paradigm can be avoided by making all authorization decisions depend only on attributes, not identities. In most cases, an assertion on the user is sufficient. ABCs allow a combination of assertions to be made, by simply showing multiple attributes (*e.g.* the user paid a subscription, and is an adult). It should be noted that in cases where the *authentication* is absolutely necessary, *e.g.* when an individual wants to communicate with one specific entity, ABCs can also be used. The ABC must simply contain an attribute representing the identity or public key of its holder. In summary, attributes achieve the same functionality as identifiers with the same security level and also provide unlinkability.

By design, the selective disclosure functionality of ABCs prevents data overcollection. Actually, an ABC can be shown (or *proved*) without disclosing any of the attributes it contains. Thus, the only leaked information is that the holder of the ABC was accredited by the trusted issuer. This information is sufficient in a scenario where resources can only be accessed by members of some institution, and that institution is the issuer of ABCs.

The multi-show unlinkability property is designed to prevent linkability of transactions. If an ABC is proved without disclosing any attribute no linking at all is possible. This is the best case scenario for the user. When showing one or more attribute(s), a user leaks some information that can be characterized in terms of k*-anonymity* [299]. For instance, consider a 30 years old male user that proves his gender and age to some service provider. From the service provider's point of view, who did not have any *a priori* information, the user could be any individual in the

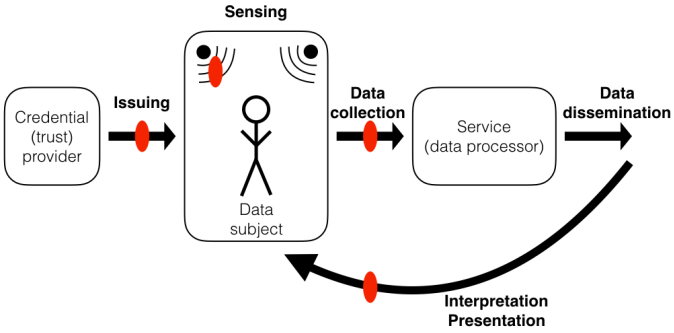

Figure 3.1: Data flow and authentication in the Internet of Things. (The red spots denote points of possible attribute-based authentication.)

context's population before the showing of attributes: the anonymity set is maximal. After learning the gender and age of the user, the service provider can restrict the potential individuals to 30 years old males in the population: the anonymity set is much smaller. More generally, when a user shows a set of attributes, it is k-anonymous among other users that have (and show) the exact same attributes, and the service provider can not distinguish between two transactions from the same user or from two users in the anonymity set. In the worst case, if the attribute is identifying the user uniquely ($k = 1$), the service provider can link of all transactions.

To model possible information privacy harms, we use Solove's taxonomy [292, 334] as a starting point. Solove discusses four groups of harmful activities with regard to privacy, from which three (information collection, processing and dissemination) are related to data flow while the fourth one is not (invasions).

As a starting point, to identify data flows where AB authentication is relevant, we again use Solove's taxonomy of privacy harms [292] (data collection, processing and dissemination). This approach is similar the work of Ziegeldorf *et al.* [334], but the authors do not address the authentication aspects of the data flow. Therefore, we develop their model further. Besides the three groups above, we consider additional activities in the IoT. First, the user is surrounded by sensors, which can be divided in two groups. Some sensors can be controlled by the users, while others not. Second, processed data is partly used by the data subject within data dissemination. Third, some part of the whole data flow is affected by credential providers that determine information access of and related to the data subject.

In Figure 3.1 the data flow and possible attribute-based authentication are represented. First, the data is sensed around the data subject. Second, some of this information is sent to the service provider. Then, the data processor, after pro-

cessing (*e.g.,* aggregate, search, analyse) the collected information, disseminates it. Finally, the access to disseminated information may be restricted to a certain group of entities (devices or people).

AB authentication can take place at four points in this model – denoted by red spots:

1. *Sensing.* User-controlled sensors, such as wearables and smart-home devices, can communicate with each other locally. In this case, the authentication can be based on attributes.

2. *Collection.* Sensors, when communicating with a service (data processor) should authenticate to the service. Possibly, the authentication can be mutual (*i.e.* the service also authenticates to the device). In both cases, it can also be based on attributes.

3. *Dissemination.* After collection and processing, data processors should communicate restricted information (*e.g.* individuals' personal information) only after AB authentication of the receiving party.

4. *Issuing.* Issuing itself is a part of the AB technology, which provides a new credential to an entity. To make sure that this entity is entitled to the particular credential, the credential provider has to authenticate the entity. This can be done with conventional identification, or by attribute-based authentication. The latter is called in [11] a dependent credential.

3.6 Use case scenario

3.6.1 Smart home

Consider a scenario involving a home owner, hereby called *user*, in possession of multiple smart devices and home appliances. The user's devices, all under his control, are connected to the Internet via a hub (installed within the home). The devices autonomously connect to remote services in order to push data to or pull data from them. Depending on their business model, some of these services may require users to get a subscription before consuming the service. Therefore, a form of authentication is necessary for the service to check if the devices belong to a valid user (*i.e.* one that paid for a subscription). In a traditional IoT setup, each user would have an account on the service's platform, where all his devices would

be registered. A service would accept requests only from devices that belong to a registered user.

3.6.2 Privacy threats

This framework carries several of the security and privacy risks presented in Section 3.4, as described in [327]. The most notable potential breach is the tracking of user among the services ecosystem *via* his *fingerprint* (based on the set of devices he owns). This fingerprint basically acts as an user identifier of pseudonym and makes all his transactions linkable, even if he does not disclose his identity. Moreover, if a user conceals his identity to a particular service S but not to another service S', S can re-identify the user with the help of S', since both services have fingerprint for that user. Together with data over-collection, common for current practices, allows for extensive user profiling. In the discussed use case profiling is performed within such a sensitive sphere of life as one's behaviour at home.

3.6.3 Applying ABC

Using ABCs, one can avoid these issues. At the same time, ABCs allow the user to have full access to the services he paid for. With respect to Figure 3.1, ABCs are mainly relevant in the issuing and collection phases, for the given use-case. After an out-of-band authentication, the user and his devices will first be issued a set of ABCs. These credentials permit the devices to authenticate to services on behalf of the user. Issuers can be various parties, including trusted third parties and the service providers themselves. ABCs can include attributes describing the model number of the device or the subscription identifier of the user.

Then, the collection phase here consists of devices making requests to the services. Before processing any such request, services require that devices authenticate by proving possession of an AB credential. When authentication succeeds, the service is assured that the devices proving to have AB credentials are genuine and belong to a valid user. At the same time neither the user nor the device identity is revealed. Information collected by the services during authentication and transaction processing includes the nature of the request itself, some meta-data such as the time and issuer identity, and the attributes disclosed by devices. We assume that the amount of information revealed by meta-data (e.g. actual IP and MAC addresses) is reduced using other privacy enhancing technologies, like anonymous communication systems [80], [103].

3.6.4 Reduced privacy risks

The attributes disclosed by devices determine how much information is revealed to a service provider. To realize data minimization, in some cases, no attributes should be disclosed at all: the simple fact that the device holds an ABC may show that it belongs to a valid user. However, other providers may supply different levels of a service depending on some additional information in the form of attributes; for example, a remote car diagnostic service may require the model or the manufacturer of the device. If service providers need this information to process requests, this information leakage cannot be prevented. The only solution would be not to use these services at all.

Disclosing the minimum set of attributes prevents the linking of a device's requests. Indeed, the multi-show unlinkability property of ABCs ensures that the only information that can be collected is that by the revealed attributes. This holds even in the worst case scenario: If the issuer and the authenticating party collude, or happen to be the same; the service providers store information about transactions centrally; and they share all this information with each other.

Chapter 4

High-speed dating: Privacy-preserving attribute matching for RFID

4.1 Introduction

RFID technology is predominantly used for identification and authentication of items and persons. In a typical setup a tag has some secret which it uses in an identification or authentication protocol with a reader. Attribute-matching protocols on the other hand focus on determining whether two or more tags have a set of attributes that match a specific relationship. By using an attribute matching protocol one can also authenticate tags by simply matching them with a tag known to be genuine. Provided both tags share an attribute (or a key) they will pass the validation.

As an important application for the tag authentication by matching we envision preventing counterfeit products. A producer can provide a reference tag to the verifier, containing the same key as the genuine products. By matching product tags with the reference one can detect counterfeits, without the key ever leaving the tags which can be protected on hardware level. Such an approach has major advantages compared to classical authentication protocols. Symmetric key authentication protocols are very efficient but require storing the secret key on the reader.

Authentication by matching combines efficiency with a typical property of asymmetric protocols that do not require the verifier to possess secret (private) keys.

Let us consider the example of a *speed-dating* party (or rather, with the protocols presented in this chapter, a *high-speed-dating* party). The typical setup of a *speed-dating* party is that singles try to find a partner through many short meetings with many people. The goal of these short conversations is to find out whether the two people share interests and want to engage in a longer conversation or a date. *High-speed dating* replaces these short match-making conversations by scanning RFID tags as follows: The organizer of the party has collected all relevant attributes (hobbies, city of residence, kids and pets preferences, etc.) of all participants in advance. Every participant receives an RFID tag which stores his or her attributes. When two persons want to decide whether it is worth starting a conversation, they just have their tags scanned simultaneously by a reader, and the reader determines whether these persons have overlapping interests and wishes. Thus, the task of a reader is to detect the fact that two participants have matches in their interests, and output the number of these matches. No false positives are desirable, since false positive will steal time of participants. The obvious target group for such a party are "nerds and geeks", who typically have very serious concerns about their privacy. They do not want a reader or another person to learn anything about them, except for the fact whether they share interests with another person or not. Also, tags' unlinkability should be preserved, so nobody can trace tags. Last but not least, no attributes stored on tags shall be disclosed. This application may not sound like the most serious scenario, but it illustrates very well what the protocol does and what properties we expect from the protocol.

4.1.1 Our contributions

This chapter presents three private-attribute-matching protocols (or *speed-dating protocols*). The first protocol uses only symmetric cryptography, the second only asymmetric cryptography, and the last a combination of both. None of the protocols requires readers to be connected to a central database; they are furthermore not required to have specific knowledge about the tags. The first two protocols provide matching for one attribute per tag. The symmetric protocol provides speed and efficiency at the cost of a lower privacy protection level. The asymmetric ones provide better privacy at the cost of a single asymmetric encryption step. Hybrid encryption can be used in the protocol that supports tags with several attributes

to further increase efficiency. All protocols provide provable security against false positives and provable privacy protection.

The remainder of this chapter is organized as follows. Section 4.2 provides preliminaries for this chapter. Section 4.3 presents the model of the system and the adversarial games. Section 4.4 introduces the lightweight symmetric protocol. Section 4.5 introduces the two remaining protocols using asymmetric primitives. Related work is discussed in Section 4.6.

4.2 Preliminaries

4.2.1 Radio frequency identification

Radio Frequency Identification (RFID) is the technology that, as its name suggests, uses electric or magnetic fields at radio frequencies to communicate identifying information. In practice though, usage of RFID technology is not restricted only to identification, but is used for access control, payment automation, verification of authenticity, etc. A branch of RFID technology operating at high frequency is Near-Field Communication (NFC). It takes advantage of a short communication distance, allowing smartphones to establish a secure wireless communication channel. What differentiates NFC from RFID is its ability to perform peer-to-peer communication. RFID specifies a more strict division of roles. The important parts of RFID are the hardware components: tags and readers. *Tags* are small chips connected to an antenna, used to receive and transmit signals. RFID tags are attached to physical objects (things) and uniquely identify these objects. *Readers* are the complementary devices that are used to communicate with tags.

Tags can be distinguished according to whether or not they have a power source. A *passive* tag does not have its own battery and receives energy from the reader. This makes it impossible for passive tags to initiate communication, but at the same time does not restrict the lifetime of the tag to the lifetime of its battery. These types of tags also have an advantage of being the cheapest to produce and are the smallest, for example, an RFID tag produced by Hitachi is 0.04 mm $\times$ 0.04 mm in size [172]. Consequently, they are very limited in their power budget that can be spent on operations and communication, and in the number of available operations to perform, as well as a very small memory size measured in Bytes. *Active* tags do have an internal power source, allowing them to run computations independently of the presence of a reader, and to initiate communication with it. The memory

size of these tags is measured in KBytes. *Semi-active* tags are the ones that use battery for performing computations, but still require power from the reader to communicate.

Readers match the type of tag they communicate with: they have to be able to follow the same protocol, and readers of passive tags emit a stronger signal than readers of active tags. The usual mode of communication between a reader and a tag is an exchange of a query, or a command and a response. When communicating with a simple tag, a reader's signals turn the tag to an "on" state to communicate whatever the tag has stored in the memory. Smarter tags distinguish different signals from a reader as different commands. Readers could be fixed like the ones used at the entrance to a supermarket, or mobile and small. Additionally, readers could perform operations and store all information locally, or they could be connected to an external database. RFID databases are designed to have a secure connection to a reader, store data related to tags and perform operations instead of readers.

Applications. Applications of RFID technology are numerous. *Identification* of objects is used in e-passports, by retailers as an anti-theft measure, in loss-resistant sport equipment like golf balls, for pets with implanted tags, etc.

Inventory management and tracking is greatly simplified with RFID tags, as they can be read automatically from the distance and in bulk. To track an object, the system can correlate the position of the reader that recently read the tag with the location of the tag (and thus the object it is attached to). Examples include inventory of items in a supermarket, books in a library, livestock on farms, runners during a race, or goods in a supply chain.

Authenticity proof allows one to verify the origin of an item it is attached to. This is often used in high-value banknotes, in the pharmaceutical industry to detect counterfeit medicine, on luxury good to detect fakes.

Matching allows to quickly verify relationships between items or people. For example, in a hospital environment new-born babies are matched to their mothers, or patients to their medicine to prevent mistakes.

RFID is often used for *access control* for buildings and rooms, in place of keys for cars and hotel rooms, and concert tickets.

Fast payment with RFID is widespread in public transportation networks like trains, trams, buses and subways. It is also used for contactless payments with credit and debit cards.

4.2.2 The game-based security model

In this section we briefly introduce the game-based approach to analyzing security of protocols. The protocols for RFID matching introduced later in this chapter will rely on a game-based security model to demonstrate their privacy-preserving properties.

Reduction. The use of the game-based proof approach started with the landmark paper by Goldwasser and Micali in 1982 [153]. The paper introduces a definition of security of cryptographic protocols by means of *reduction*. Reduction is done by proving that the problem on which a protocol is built is equivalent to one of the well-studied problems in computational complexity theory that are assumed to be hard. Hardness, in this sense, means that there is no algorithm that would solve the problem in polynomial time, thus, for a *computationally restricted adversary*, solving such a problem is infeasible. A paper by Lenstra et al. [209] proposes the following intuitive understanding of infeasibility: to break a system with 140-bit security one would need as much energy as the sun produces in a year.

The goal of the reduction is to demonstrate, that in order to break the protocol, an adversary would have to be capable of solving the computationally infeasible problem. For example, factoring of large integers is a well-recognized hard problem, which the best algorithms currently known take sub-exponential time to solve. The usual approach is to show that if there is an adversary that can break the scheme, another adversary can use him as a subroutine to solve the hard problem. Alternatively, one can demonstrate that an adversary would have to be able to break cryptographic primitives used in the protocol that are already already proven to be secure in order to be able to break the protocol itself.

Games. The concrete properties (or security notions) that the cryptographic system provides are formulated as a game between an adversary and a *challenger*. For example, the property usually required of encryption schemes is *indistinguishability* (IND), introduced in [153]. Informally, indistinguishability ensures that an attacker is unable to extract any information about the plaintext of an encrypted message [33]. Games have strictly specified phases prescribing interaction between players and possible actions an adversary can perform. Often the phases include: setup, execution and challenge. During the setup phase, system is initialized and all the required parameters are generated. The execution phase primarily prescribes the actions an adversary can perform and the information he can learn. The challenge phase fo-

cuses on an output by an adversary, which is used to establish whether he won or not.

Conditions to win. Conditions to win are explicitly stated at the end of the game description and can be defined by two different approaches: a *concrete approach* or an *asymptotic approach*. The concrete approach specifies the *advantage* (in other words, the success probability) for an adversary to break the scheme to be at most ϵ for a specified amount of time. Time is often measured in processor cycles or the number of operations an adversary performs.

The asymptotic approach specifies the advantage of an adversary and his running time as a function of the security parameter λ chosen during system initialization. For example, the security parameter of a pseudo-random number generator could be the length of its seed, and for an encryption scheme it could be the length of its key. The running time of an adversary is restricted to be polynomial in λ, and his advantage has to be *negligible* in λ. The advantage $\epsilon(\lambda)$ is said to be *negligible* if for all $c > 0$ there exists $K > 0$ such that $\epsilon(\lambda) < n^{-c}$ for all $n \geq K$.

Security notions. To capture the way an adversary can interact with the system and what information he can receive, one introduces cryptographic *oracles*. An oracle acts as a black-box that receives queries and provides answers according to its description. Usually the game specifies different oracles to which an adversary has access in different phases of the game. The difference in which kinds of oracles an adversary can access distinguish different types of attacks. The most well-known types of attacks are attacks on encryption systems: the *chosen-plaintext attack* (CPA), the *non-adaptive chosen-ciphertext attack* (CCA1), and the *adaptive chosen-ciphertext attack* (CCA2).

In the CPA game an adversary is provided with access to an encryption oracle (or a public key in an asymmetric encryption system), that replies with the ciphertext of the plaintext messages sent from the adversary. During the challenge phase the adversary submits two plaintexts and gets back an encryption of one of them. His goal is to output the correct guess as to which of the two plaintexts was encrypted. In CCA1 attack, the adversary can additionally access the decryption oracle until the challenge phase starts. CCA2 is equivalent to CCA1, except that it also allows access to the decryption oracle during the challenge phase, having only one restriction - the queries to this oracle must not include the challenge ciphertext itself. The change from CCA1 to CCA2 captures the ability of an adversary to adapt his attack depending on the challenge ciphertext received.

Let us depict the game of an *IND-CCA2* property. Let the encryption with the key ke of the message m be denoted by $c = E_{ke}(m)$, decryption with the key kd correspondingly by $m = D_{kd}(c)$, and the key generation algorithm be denoted as G. Let ϵ be negligible with respect to λ, and an adversary be denoted as A.
IND-CCA2 notion is defined as the following game $\textbf{Exp}_A^{IND-CCA2}(\lambda)$:
Setup: $ke,\ kd = G(\lambda)$
Learning: The adversary may perform calls to the decryption oracle to obtain plaintexts of submitted ciphertexts.
Challenge:

1. The adversary chooses m_0, m_1 such that they have the same length and $m_0 \neq m_1$.
2. The adversary sends m_0, m_1 to the challenger.
3. Let $b \in_R \{0, 1\}$. The challenger returns $c_b = E_{ke}(m_b)$ to the adversary.
4. The adversary is allowed to perform calls to the decryption oracle with any $c \neq c_b$.
5. The adversary outputs a guess bit b'.

Result: This experiment outputs true if the adversary correctly outputs $b' = b$.

A scheme is IND-CCA2 secure if the advantage of the adversary of winning the game is negligible:

$$Adv_A^{link}(\lambda) = |Pr[\textbf{Exp}_A^{IND-CCA2}(\lambda) = true]| < \frac{1}{2} + \epsilon\,.$$

The Random Oracle model. So far we described proofs that rely only on one kind of assumptions - the well-studied problems from computational complexity theory that are believed to be infeasible to solve in polynomial time. This is called the *standard model*, sometimes referred to as the bare or plain model. In some cases one needs to make more assumptions, such as replacing some cryptographic primitives with ideal versions in order to be able to perform the proof.

The Random Oracle (RO) model is a well-known example of a non-standard model. It was introduced by Bellare and Rogaway [37] who argued for the practicality of ROs already used in the various previously published papers. They proposed an approach: to use a publicly accessible RO in place of a certain primitive, usually a hash function. An RO is an oracle that uses a function chosen uniformly from all possible functions for each of its outputs, except for repeated queries. When RO

receives the same input, it repeats its previous output. Because no hash function in practice is an RO, proof in the RO model is considered weaker than proof in the standard model. There are also examples of systems secure in the RO model, but losing their security, when the RO is substituted with an existing primitive [76], [31], [239]. Koblitz and Menezes repeatedly point out that these schemes are artificial examples made without following "sound cryptographic practice" [199], [198]. Even further, in the same papers they argue that the fact that the best researchers were so far only able to come up with such counterexamples serves as evidence in favor of using the RO model.

Security guarantees and their limitations. There are several limitations one has to keep in mind when talking about cryptographic schemes with proven security. Firstly, reductionist proofs are based on assumptions, and if these assumptions fail no security can be provided. Secondly, security guarantees are defined as a game, the implications of which are sometimes not all immediately obvious and need close consideration. Thirdly, only a limited number of well-known attacks are covered in a proof. An example of an attack class not covered in a reductionist proof is the class of side-channel attacks. These are attacks that rely on the implementation specifics of a cryptographic protocol, for example dependencies of the time spent on operations depending on the values of the key. Finally, one has to separately take care of software bugs and incorrect uses of the protocols, for example the failure to generate a fresh random nonce for signatures by Sony [40].

4.2.3 Cryptographic primitives

One of the important primitives used in speed-dating protocols is a pseudo-random function (PRF), which cannot be efficiently distinguished from a truly random function. In the game definition of a PRF, an adversary submits inputs to the PRF challenger. The challenger replies with either an output of the PRF, or an output of a truly random function. The adversary wins if he has a non-negligible advantage to distinguish these two possible outputs. Let a secure PRF be denoted by $Fun_k(\cdot)$, where k is the key of the PRF function.

Note that we can efficiently construct a PRF from a keyed hash function (HMAC), through the Merkle-Damgård iteration [226], if keyed using the input as described, for example, in [69, Section 6]. In a similar way, one can construct a PRF by using keyed modes of lightweight sponge-based hash functions like Quark [19] and Kec-

cak [254] to construct a PRF function. A cryptographic hash function is denoted by $H(\cdot)$ further in the text.

It was proven by Bellare et al. [36, 35] that any PRF is a secure message authentication code (MAC). This property is essential for our protocols. Let us introduce *secure MACs* in a form of brief game description, for details see [152]. During the MAC-unforgeability game, an adversary queries the challenger with distinct messages and obtains MACs for them. He can also submit several message-tag pairs to the verification oracle. An adversary wins, if he succeeds, with a non-negligible advantage, in outputting a valid message-tag pair, not previously requested from the challenger.

Both encryption schemes used in the speed-dating protocol are required to have the *IND-CCA2* property. Let a symmetric encryption scheme be denoted by $symENC = (G', E', D')$ and an asymmetric encryption scheme by $pkENC = (G, E, D)$.

4.3 Model and notations

Let C be the set of all tags in the system. Each tag $a_j \in T$, $i \in \{1, 2, \ldots, n\}$ is supplied with attributes. Each attribute is a human-readable string of arbitrary length, the set of all attributes in the system is denoted by $C = \{a_1, \ldots, a_p\}$. Let the security parameter be λ, and the number of attributes and tags be polynomially bounded in λ. Each attribute in the system is related to a secret key stored on a tag in the following way. An issuer starts with an attribute set C. To setup the system, the tag issuer generates a set of *keys* $C = \{k_1, \ldots, k_p\}$, which are each associated with an attribute. Each key k_j for $j \in \{1, \ldots, p\}$ is a λ-bit string.

Each tag stores a subset of K of cardinality at most m, where $1 \leq m \leq p$. For two tags t_i and t_j that share an attribute a_j the corresponding key for this attribute is stored on both tags. For simplicity, further in the text we use the term keys and attributes interchangeably.

The goal of the protocol is to determine the number of attributes on two tags that match. Let the state S_i denote the set of attributes stored on the tag t_i. Note, that if tags t_i and t_j have the very same attributes assigned to them, their states are equal $S_i \equiv S_j$. This state is assigned by the issuer during the setup phase: a function $Setup(\lambda)$ is used to assign state S_i to the tag t_i, with $i \in \{1, 2, ..., n\}$, and generates keys for readers (if necessary). All secret values are generated taking the security parameter λ into account. Later, during the protocol run, a reader R

simultaneously scans two tags t_i and t_j to obtain the result of the function $Match$: $T \times T \rightarrow \mathbf{N}$. This function computes the cardinality of the intersection of the states of two tags: $Match(t_i, t_j) = |S_i \cap S_j|$. Some applications do not require $Match$ to compute the cardinality of $S_i \cap S_j$, but only need to know whether this intersection is empty or not. For those applications we use $Match : T \times T \rightarrow \{0, 1\}$.

The function $Match$ must fulfill the following properties:

1. *Correctness*: In the absence of adversaries the output is correct.

2. *Unforgeability*: False positives are impossible, that is an adversary is unable to convince a reader that tags match on more attributes than they actually are.

3. *Unlinkability*: Neither a reader, nor an external adversary is able to decide, whether in two protocol runs the same tags participated twice or not.

4. *Confidentiality*: After a protocol run nobody can learn any of the attributes (corresponding keys) stored on a tag, unless possessing a valid key for an attribute. The amount of attributes stored on tags is computationally hard to derive from a protocol run, unless possessing all keys in the system.

The protocols presented in this chapter do not protect against false negatives. Thus, they are useful for applications that do not require to prevent false negatives. To summarize, the system has the following functions:

1. $Setup(\lambda)$: is used to generate a private key or a public/private key pair for a reader (if specified by the protocol) based on the security parameter λ. It assigns a state S_i to all tags $t_i \in T$. The state includes the set of secret keys $k_{i1}, k_{i2}, ..., k_{im}$ assigned to the tag and the publicly known information (e.g. public keys of readers).

2. $Match(t_i, t_j)$: Is a protocol carried out by two tags t_i and t_j, and a reader R. The protocol is initiated by the reader. As a result of the protocol, the reader obtains the cardinality of the intersection of S_i and S_j.

4.3.1 Adversary model

The security of our protocols relies on the secrecy of the keys stored on tags. We thus make the common assumption that those keys are stored in a secure way, and that computations involving those keys are implemented in a way that does not leak information about the keys (for example, through side channels). The adversary

controls all the communication, pretends to be one of the valid tags, but does not perform relay attacks using a tag outside the proximity of the reader. The reader is assumed to behave according to the protocol. Thus, it is considered "honest but curious".

The type of an adversary A is specified by the actions he can perform. Let π be a protocol execution entity. The oracles below define the whole set of possible actions. An adversary gets an access to a subset if oracles depending on his type. Oracles distinguish between the left and the right message denoted by m_{left}, m_{right}. This notion is needed to distinguish communication with tag of the left and right side. The oracles are:

- $Launch(m_{\text{left}}, m_{\text{right}}) \rightarrow \pi, m$: when this oracle is called, the reader starts a new protocol execution π by sending out the message m. The whole execution of the protocol can then be performed using oracles $SendReader$ and $SendTag$. These two oracles can be used to simulate the $Execute$ oracle from the model defined by Juels and Weis [188]

- $SendReader(m_{\text{left}}, m_{\text{right}}, \pi) \rightarrow (m'_{\text{left}}, m'_{\text{right}})$: sends a message m to a reader from the left side (right side or both) in the context of protocol execution π. The output of the oracle is a response of the reader m' sent in any of the directions according to the description of the protocol.

- $SendTag(m, t_i) \rightarrow m'$: sends a message m to a tag t_i. The output of the oracle is a response of the tag m'.

- $Result(\pi) \rightarrow x$: outputs the result of function $Match$ after the protocol execution π.

- $Corrupt(t_i) \rightarrow s_i$: returns the internal state of the tag, allowing an adversary to learn all secret keys stored on this tag.

The model and the privacy game presented in this section are inspired by work of Juels and Weis [188], Gildas [20], Vaudenay [311] and Hermans et al. [165]. Unfortunately, all these models were designed with the classes of protocols in mind that are different from our protocol. All the models consider the scenario of communication between a reader and a single tag. We therefore adopt the classification by Vaudenay and modify the Juels and Weis game for unlinkability to fit the needs of matching protocols[1].

[1] Later we found a model by Gildas Avoine [21] that allows for a reader to communicate with several tags.

The classification by Vaudenay is faceted in two dimensions. An attacker who does not have access to the $Result$ oracle is called *NARROW*. An attacker who does is called *WIDE*. An attacker who cannot corrupt tags is called *WEAK*. An attacker without any restrictions regarding corruption of tags is called *STRONG*. An attacker who is not allowed to perform any protocol interactions after he corrupted one tag is called *FORWARD*.

4.3.2 Unforgeability

The goal of an adversary is to convince a reader that the number of matching attributes is larger than it actually is. We call a protocol *unforgeable* if it resists this attack. Let S be a system and A be an adversary.
Unforgeability is defined as the following game $\mathbf{Exp}^{forge}_{S,\,A}(\lambda)$:

Setup: $Setup(\lambda)$ is used to initialize all readers and tags.
Learning: The adversary may perform calls to the available oracles on the given set of tags T. The set of available oracles depends on the adversary type. The strongest adversary gets access to: $Launch$, $SendReader$, $SendTag$, $Corrupt$. Let the union of the sets of all corrupted tags be C_t.
Challenge:

1. The adversary chooses a tag t_i, to which he did not call a $Corrupt$ oracle.
2. The tag t_i is removed from the set T. The challenger returns t_i to the adversary.
3. The adversary is not allowed to modify any of the messages sent to or from the tag t_i. The adversary is simulating a tag t_j on the other side.

Result: The experiment outputs true if the reader outputs a value lager than $|S_i \cap (S_j \cup C_t)|$.

The advantage of adversary of winning the game is defined as:

$$Adv^{forge}_{S,\,A}(\lambda) = Pr[\mathbf{Exp}^{forge}_{S,\,A}(\lambda) = true]\,.$$

We call the system unforgeable if a maximal advantage of all polynomial time adversaries is negligible in the security parameter λ. During the challenge phase an adversary can only passively eavesdrop messages exchanged between the challenge tag and a reader. The adversary has to simulate the other tag, which will be matched by the reader with the challenge tag.

The model above considers tag corruption by taking into account that the keys extracted from corrupted tags will trivially allow the adversary to increase the match count output. For the protocols presented in this chapter only WEAK adversaries are considered for unforgeability, and hence $C_t = \emptyset$.

4.3.3 Unlinkability

The goal of the attacker A is to distinguish tags, thus breaking their unlinkability. In case an attacker is able to obtain the result of the protocol run, an attack on unlinkability becomes trivial as pointed out in [123]. It is sufficient to have one tag participate in two protocol runs with potentially different tags. By comparing the result (i.e. cardinality of the intersection) one can determine if that tag was matched against different tags or not. That implies that the $Result$ oracle cannot be used by an attacker. The match protocol by its nature is giving away information about tags, namely the relationships between them. There are two approaches for designing the speed dating protocol to tackle this problem. One is to make sure that the protocol itself is not providing any evidence of the relationships between tags, except for the output of the reader. The other is to provide only a $Minimal$ level of protection. In this minimal model an attacker is unable to recognize the same tag he was observing before, once he initiates a protocol with only this tag. To illustrate, assume an attacker was collecting interactions among tags on the speed dating party. After the party is over, he suddenly sees a person wearing an RFID tag from this party. An attacker triggers a protocol, having no other valid tag at hand. He should be unable to learn the identity of the tag even having all the old protocol run transcripts at hand.

Unlinkability of an attacker A in the system S is defined as the following game $\mathbf{Exp}^{link}_{S,A}(\lambda)$:

Setup: $Setup(1^\lambda)$ is used to initialize all readers and tags.

Learning: The adversary may perform calls to the available oracles on the given set of tags T. The set of available oracles depends on the adversary type: $Launch$, $SendReader$, $SendTag$, $Corrupt$.

Challenge:

1. The adversary chooses two tags t_i and t_j, to which he did not call a $Corrupt$ oracle.

2. The challenger assigns $t_0^* = t_i$ and $t_1^* = t_j$. Both tags are removed from the set T.

3. Let $b \in_R \{0, 1\}$. The challenger returns t_b^* to the adversary.

4. The adversary is allowed to perform calls to the oracles: $Launch$, $SendReader$, $SendTag$, having tag t_b^* on one of the sides and any of the tags from the set T on the other.

5. The adversary outputs a guess bit b'.

Result: The experiment outputs true if the adversary correctly outputs $b' = b$.

The advantage of adversary of winning the game is defined as:

$$Adv_{S,A}^{link}(\lambda) = |Pr[\mathbf{Exp}_{S,A}^{link}(\lambda) = true] - \frac{1}{2}|\,.$$

We call the system unlinkable if the maximal advantage of all polynomial time adversaries is negligible in the security parameter λ.

$Minimal$ privacy is achieved if an attacker during challenge phase gets only one tag t_b^* to communicate with and he has to simulate a tag on the other side. This game modification is used only against a *WEAK* adversary, if the adversary has an access to the $Result$ oracle. Otherwise an attacker could simulate all the tags he corrupted during the learning phase. Thus there would be no difference between two games, since an attacker could use all the broken tags to let them communicate with the challenge tag. This would allow an attacker to use the knowledge of topology of tag's relationships to win the game.

4.4 One-key symmetric speed dating

The protocol presented in this section prevents false positives and provides minimal privacy. False positives occur when a tag does not possess any attribute matching with attributes on the other tag. And yet it manages to make a $Match$ function output that tags have a match. Minimal privacy is the privacy protection achieved against an adversary that can read output of the protocol runs and thus build a topology of tag's relationships. The protocol only requires a few calls to a (lightweight) hash functions.

4.4.1 Single attribute per tag

We start with the setting that each user has a single attribute. Assume tags named (for convenience) by their owners, namely Alice and Bob. Tags Alice and Bob possess respectively keys k_A, $k_B \in K$. These keys are representing attributes of tags. A reader scans both tags to figure out whether their attributes match or not. Figure 4.1 depicts the protocol.

The general idea of the protocol is the following:

1. *Commit phase*. Tags generate random numbers and exchange commitments with each other. These commitments later on help a reader to identify replies of tags and prevent cheating. The exchange is happening with the help of the reader.

2. *Check phase*. Tags create pseudo-random values from the challenges by feeding them to a PRF function Fun. These values are exchanged with the help of a reader. Tags perform a check of the values generated by the other tag using their secret keys. After this phase tags know whether they have an equal key or not.

3. *Match phase*. If there is a match, tags open their commitments towards the reader, which determines the result of the protocol.

Commit phase

1. The tags generate random values and calculate commitments to these values. Alice generates: $r_A \in_R \{0, ..., 2^\lambda - 1\}$, $c_A = H(r_A)$. Bob generates: $r_B \in_R \{0, ..., 2^\lambda - 1\}$, $c_B = H(r_B)$.

2. The tags send the commitments c_A, c_B to a reader.

3. The reader checks if $c_A = c_B$. If so, the protocol run is terminated with output $\perp$. The reader forwards c_B to Alice and c_A to Bob.

4. Each tag concatenates the commitments and puts the value it generated on the last position. Alice, for example, obtains $c_B||c_A$.

5. Each tag computes the PRF function Fun using their group keys and sends it to the reader. Alice computes: $ch_A = Fun_{k_A}(c_B||c_A)$. Bob computes: $ch_B = Fun_{k_B}(c_A||c_B)$.

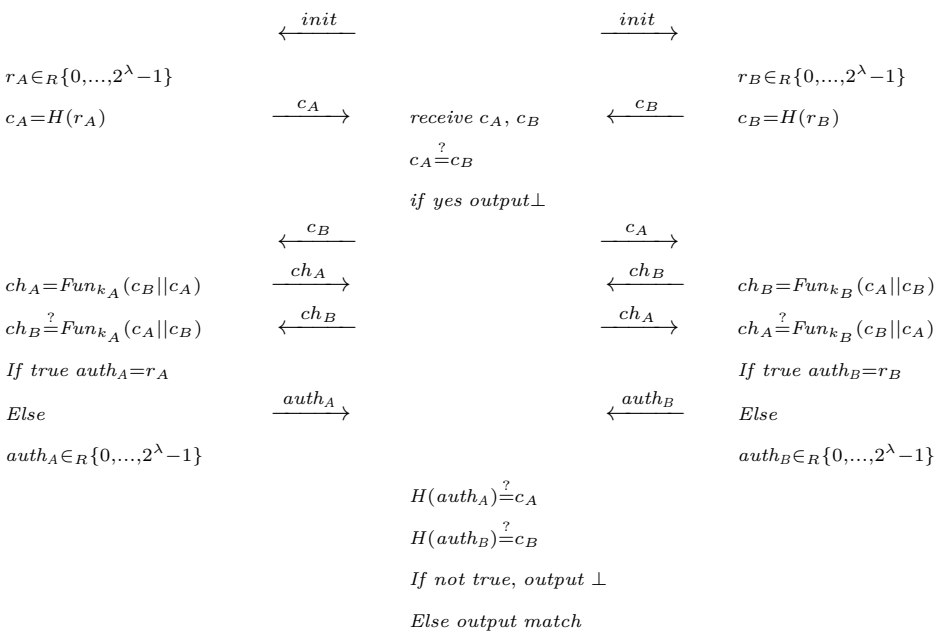

Figure 4.1: One-key symmetric speed dating protocol.

6. The reader forwards ch_A to Bob, ch_B to Alice.

Check phase

1. Each tag checks the received commit value. Alice checks $ch_B \stackrel{?}{=} Fun_{k_A}(c_A||c_B)$. Bob checks $ch_A \stackrel{?}{=} Fun_{k_B}(c_B||c_A)$.

2. If the equality holds, Alice computes: $auth_A = r_A$. Else, she sends the response with $auth_A$ filled with a random value. Similarly, Bob computes: $auth_B = r_B$ if equality holds. Else, he sends the response with $auth_B$ filled with a random value.

3. The tags send $auth_A$ and $auth_B$ to the reader.

Match phase

1. The reader checks that $H(auth_A) \stackrel{?}{=} c_A$ and $H(auth_B) \stackrel{?}{=} c_B$. If any of these two values are false, the reader outputs $\perp$, otherwise it outputs a match.

Theorem 1. *If Fun is a PRF, then $Adv^{forge}_{S,A}(\lambda)$ of a WEAK adversary, that runs in polynomial time, to win the unforgeability game is negligible in the random oracle model.*

Proof. Since the protocol is symmetrical, we consider the protocol from the perspective of tag Alice without loss of generality. Since any PRF is a secure MAC [36], [35], we can consider Fun to be a secure MAC function. Assume an adversary A can break the protocol unforgeability. We show that there exists an adversary A', that can then win the MAC unforgeability game using adversary A as an oracle.

The hash function is modeled as a random oracle RO. The adversary A' is interacting with a MAC game challenger possessing a key k_{ch}. Adversary A' is simulating the unforgeability game for an adversary A by answering all requests an adversary A makes to oracles with a small exception. One particular tag t_i (or more) however is simulated with a help of the MAC challenger. This tag possesses a key k_{ch}. Let us call the corresponding attribute a_{ch}. This tag is simulated to A by A' in the following way:

1. First $SendTag$: A' outputs $c_A = RO(r_A)$ according to the protocol.
2. Second $SendTag$: A' provides $c_B||c_A$ as an input to the MAC challenger, and returns it's output as ch_A.
3. Third $SendTag$: A' validates the input value ch_b. If ch_b was generated by the MAC challenger, then A' knows the values should match (this can be double-checked with the help of the MAC verification oracle). Otherwise A' directly knows what the result should be, as he is simulating the rest of the tags in the system. A' returns an output, as specified by the protocol.

During the learning phase of the adversary A, A' gets to see different tuples of messages and corresponding tags: $m = (c_A||c_B), t = MAC_{k_i}(c_A||c_B)$, MAC value generated by a challenger. Since r_A and hence c_A is selected randomly, there will be no repeating values with overwhelming probability. Additionally, since H is a cryptographic hash function, A cannot win the unforgeability game by finding preimages of commitments c_A with overwhelming probability.

Assume A selects a tag with key k_{ch} as challenge tag. Assume the challenge tag is Bob and the adversary takes the role of Alice. Note that communication between the reader and the challenge tag is performed directly by the challenger. If A wins the unforgeability game, it has to produce a valid ch_A (otherwise r_B will not be sent to the reader and hence validation will fail). This ch_A will be the MAC of an input

$c_A||c_B$ that was not used previously in the game, since c_B is fresh. Hence, ch_A can be forwarded to the MAC challenger to win the MAC unforgeability game.

The probability that the adversary selects the tag t_i is at least $\frac{1}{n}$. If the non-negligible advantage of A to win the game is ϵ, and the cardinality of the set of tags is n, then the advantage of A' is $\geq \frac{\epsilon}{n}$. This value is non-negligible, since n is polynomially bounded in a security parameter λ (see Section 4.3).

□

Theorem 2. *If Fun is a PRF, then $Adv^{link}_{S,A}(\lambda)$ of a WEAK adversary, that runs in polynomial time, to win the minimal unlinkability game is negligible in the random oracle model.*

Proof. Since the protocol is symmetrical, we consider the protocol from the perspective of tag Alice without loss of generality. We are going to show that an attacker is unable to distinguish any of the challenge tags from a simulator that is returning random values, and, thus, is unable to distinguish challenge tags themselves.

The hash function is modeled as a random oracle RO. We now simulate the $SendTag$ oracle as follows to an adversary, explaining each step of the protocol:

1. A tag generates fresh pseudo-random values r_A and returns $RO(r_A)$.
2. Upon receiving c_B, the simulated tag returns a random value as ch_A.
3. Upon receiving ch_B, the simulated tag outputs a random value. Since the adversary is playing the minimal game, there is no other valid tag to create the proper ch_B value. Next, an attacker cannot forge a value output by Fun, as proven by Thm. 1. Thus, ch_B can never be accepted as valid.

The above simulated tag is indistinguishable from a real tag. First, the random ch_A is indistinguishable from $Fun_{k_A}(c_A||c_B)$ since Fun is a PRF, r_A is selected randomly and c_A is never repeated as an output of RO. Second, when replying to ch_B we can rely on the soundness of the protocol, as proven by Theorem 1. This ensures that it is impossible for an adversary to forge ch_B. So either it was sent by a tag (and hence a match will be found) or it was forged and should be rejected by the tag. Finally, since the game is minimal, an attacker does not have any possibility to distinguish between the real tag and the simulator by matching them with other tags and comparing the output.

Assume the challenge bit $b = 0$. Tag t_0 is indistinguishable from a simulated tag. The same argument applies to challenge bit $b = 1$. Hence, an attacker cannot distinguish between two tags.

□

4.5 Match protocol for asymmetric encryption

The protocols presented in this section prevent false positives and provide a higher privacy level under the following requirement. An adversary must be unable to obtain the output of the protocol runs and, thus, build a topology of tag's relationships. The cost of the higher privacy level is the usage of public-key encryption.

4.5.1 One-key asymmetric speed dating protocol

The advantage of the protocol in this section is that it protects confidentiality of the exchanged messages. Thanks to that, when an adversary corrupts tags and obtains their secret keys, it will not help him to succeed in identifying tags. This also implies that any external observer is unable to learn the result of the protocol from the exchanged messages. Also, the protocol is easily expandable to handle the case of tags storing multiple attributes.

Assume each user can possess a single attribute as in the previous section. Tags Alice and Bob posses respectively group keys k_A, $k_B \in K$. An asymmetric encryption system $pkENC$ is used in the protocol. A reader holds a public-private key pair (pk, sk), all tags are supplied with the public key pk of a reader.

The asymmetric protocol can be obtained from the symmetric one (Section 4.4) in two steps. The first step is to encrypt messages sent between a reader and each tag. To ensure that ciphertexts of two tags differ, tags append the random number to the Fun value before encrypting. The second step is to provide the same inputs to the Fun function on both tags, since there is no need to produce different outputs of the PRF function. The reason is that messages appended with unique randomness are sent encrypted. This way the protocol is protected from trivial replay attacks, and the need for commitments c_A, c_B and their openings is eliminated. Figure 4.2 illustrates the protocol.

Speed dating protocol:

1. Tags generate random values and send c_A, c_B to the reader. Alice generates: $c_A \in_R \{0, ..., 2^\lambda - 1\}$. Bob generates: $c_B \in_R \{0, ..., 2^\lambda - 1\}$
2. Reader checks if $c_A = c_B$, then protocol run is terminated with output $\perp$. Otherwise the reader exchanges random numbers between tags.

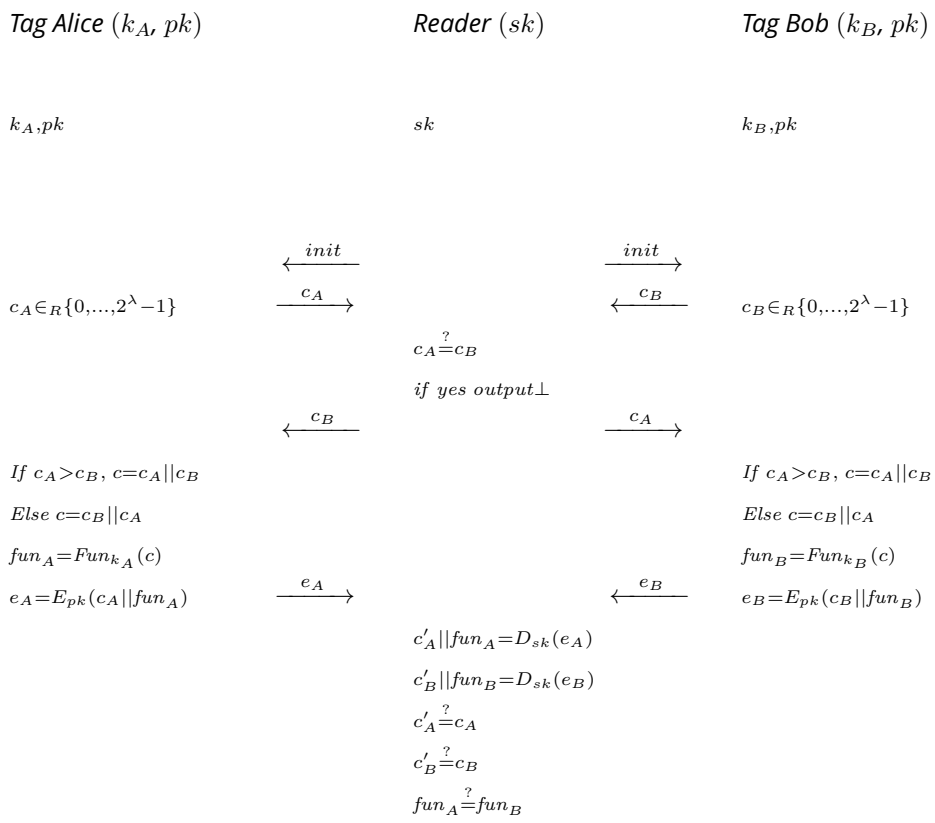

Figure 4.2: One-key asymmetric speed dating protocol.

3. Tags sort random numbers. Assume, that $c_A > c_B$. Alice learns that she has to put c_A in the beginning. Bob learns that he has to append c_B to the end.

4. Tags compute Fun values using their group keys. Alice computes: $fun_A = Fun_{k_A}(c_A||c_B)$. Bob computes: $fun_B = Fun_{k_B}(c_A||c_B)$

5. Tags send Fun values $E_{pk}(c_A||fun_A)$ and $E_{pk}(c_B||fun_B)$ over secure channel.

6. Reader decrypts received values using his secret key sk.

7. Reader checks if the decrypted messages contain c_A and c_B values.

8. Reader checks if $fun_A \stackrel{?}{=} fun_B$. If true reader outputs *Match*. Otherwise it outputs $\perp$.

Theorem 3. *If the encryption scheme $pkENC$ is IND-CCA2 secure, then $Adv^{forge}_{S,A}(\lambda)$ of a WEAK adversary, that runs in polynomial time, to win the unforgeability game is negligible in the random oracle model.*

Proof. Assume an adversary A can break the protocol unforgeability. We show that there exists an adversary A', that can then win the IND-CCA2 game using adversary A as an oracle.

The adversary A' is interacting with a IND-CCA2 game challenger possessing a secret key sk. Adversary A' is simulating the unforgeability game for an adversary A by answering all requests an adversary A makes to oracles. The simulation to A by A' is performed in the following way:

1. All the tags are fully simulated by A', who possesses all the corresponding keys of tags.

2. A' publishes pk of the IND-CCA2 game challenger as a public key of a reader. This, together with the previous item allows A' to answer $SendTag$ queries.

3. Since A is WEAK, no queries to $Corrupt$ oracle are accepted.

4. Consider the case when A initiates a protocol run without having a valid tag on one of the sides. To answer the $Result$ oracle query, A' needs to know the plaintext message encrypted by an adversary using pk and resulting in the ciphertext e. For this, A' will submit the ciphertext e to the decryption oracle within the IND-CCA2 game.

After the learning phase of unforgeability game is over, A chooses some tag t_i for the challenge phase according to the game definition. Let t_i be on the side of Alice. When A initiates the protocol execution, A' outputs on behalf of t_i a random value c_A according to the protocol and receives c_B from A. Further, A' computes the value fun_A according to the protocol. After that, A' computes two challenge messages for the IND-CCA2 game. The first message is constructed as $m_0 = c_A||fun_A$, the second message is $m_1 = c_A||ran$, where value ran is the value generated at random. As a response, A' receives an encryption of either m_0 or m_1, which he outputs for A as e_A in the unforgeability game.

Since A can can perform successful forgeries, he can output such e_B that the reader will output "match". For this e_B needs to be decrypted using sk into a message $c_B||x$, where x matches either ran or fun_A, depending on the plaintext of e_A. Now, A' submits e_B to the decryption oracle within the IND-CCA2 game. As a result

he gets either $c_B||fun_B$, where fun_B is equal to fun_A, or $c_B||ran$, which helps him to learn if he received earlier an encryption of m_0 or m_1. Thus, A' is able to win the IND-CCA2 game.

□

Theorem 4. *If the encryption scheme $pkENC$ is IND-CCA2 secure, then $Adv^{link}_{S,A}(\lambda)$ of a (non-minimal) NARROW-STRONG adversary, that runs in polynomial time, to win the unlinkability game is negligible in the random oracle model.*

Proof. Since the protocol is symmetrical, we consider the protocol from perspective of tag Alice without loss of generality. The goal of the proof is to show that if an adversary A can break unlinkability of the protocol, the adversary A' can win IND-CCA2 game. The adversary A' interacts with the IND-CCA2 game challenger, possessing a key pair (pk, sk). The adversary A' simulates the unlinkability game for an adversary A by answering all requests A makes to oracles. The reader key pair in the simulation is the key pair of the IND-CCA2 game challenger.

During the learning phase A' answers all the oracle queries from A' using his knowledge of tags' keys and the value of pk. To get decryptions of the messages encrypted with pk, A' queries the IND-CCA2 oracle. During the challenge phase A' creates messages for both challenge tags $c_A||fun_{A0}$ and $c_B||fun_{A1}$. A' then submits both messages to the IND-CCA2 challenger. The received ciphertext $E_{pk}(c_b||fun_{Ab})$ is forwarded to A. If A can break unlinkability of the protocol, thats is distinguish between tags, A' will be able to distinguish which message was encrypted.

A NARROW-STRONG adversary cannot get the result of the protocol run, but he can corrupt tags. Corrupting tags will provide an adversary with secret keys of tags. However, it does not help him in distinguishing tags and their output. This holds for the simple reason, all the information related to tags is transferred encrypted using the asymmetric IND-CCA2 encryption scheme Enc. This implies, encrypted messages do not provide an adversary with any useful information.

□

Theorem 5. *If Fun is a PRF, then $Adv^{link}_{S,A}(\lambda)$ of a polynomial-time adversary that possesses the private key of a valid reader to win the minimal unlinkability game is negligible in the random oracle model.*

Proof. Since the protocol is symmetrical, we consider the protocol from the perspective of tag Alice without loss of generality. We are going to show that an attacker is unable to distinguish any of the challenge tags from a simulator that is returning random values, and, thus, is unable to distinguish challenge tags themselves.

We now simulate the $SendTag$ oracle to an adversary as follows:

1. Upon getting initialization request, a fresh random value c_A is output.
2. Upon receiving c_B, an encryption with pk of a random value $c_A||x$ as e_A is output.

The above simulated tag is indistinguishable from a real tag. The first output is generated the same way as in the real tag. After an attacker decrypts the second output of a simulated tag, he obtains $c_A||x$. The random value x is indistinguishable from $Fun_{k_A}(c_A||c_B)$ since Fun is a PRF. The value c_A is a selected randomly. Finally, since the game is minimal, an attacker does not have any possibility to distinguish between the real tag and the simulator by matching them with other tags and comparing the output.

Assume the challenge bit $b = 0$. Tag t_0 is indistinguishable from a simulated tag. The same argument applies to challenge bit $b = 1$. Hence, an attacker cannot distinguish between two tags.

□

4.5.2 Many-key asymmetric speed dating protocol

Assume each user can possess at most m attributes. Tags Alice and Bob possess respectively group keys $s_A = \{k_A[i] \in K \cup \{\perp\} | i \in \{1, \ldots, m\}\}$, $s_B = \{k_B[i] \in K | i \in \{1, \ldots, m\}\}$. All tags are supplied with the public key pk and all readers possess the private key sk.

This protocol can be obtained from the one-key version by handling several attributes instead of one. An important change is required due to the necessity to hide the amount of attributes on each tag. Tags are generating $f_A[i]$ and $f_B[i]$ values using their keys, $i \in \{1, \ldots, m\}$. Whenever a tag has less than m attributes, the f values are filled with randomness. Tags perform random permutations on f values before they are sent to a reader. This is done in order to conceal the order of attributes, otherwise this could expose sensitive information about tags to the reader.

Upon obtaining and decrypting all of the $f_A[i]$ and $f_B[i]$ values from two tags, a reader starts by sorting both sets descending. After that a reader can easily compute an intersection of two sets Int and outputs its cardinality $|Int|$. Figure 4.3 depicts the protocol.

The efficiency of this protocol can be substantially improved by changing from using only asymmetric encryption scheme to using a hybrid one. In that case an

asymmetric encryption is used to securely transfer key material. A hash function is applied to it as a key derivation function. The result is used as a key for a semantically secure encryption system.

Theorem 6. *If the encryption scheme $pkENC$ is IND-CCA2 secure, then $Adv^{forge}_{S,A}(\lambda)$ of a WEAK adversary, that runs in polynomial time, to win the unforgeability game is negligible in the random oracle model.*

Proof. The proof is similair to the one in the Theorem 3. It differs in the following.

A' computes two challenge messages for the IND-CCA2 game. The first message is constructed as $m_0 = c_A||set_A$, where set_A is constructed according to the protocol. The second message is $m_1 = c_A||set_{ran}$, where each value of the set is the value generated at random. As a response, A' receives an encryption of either m_0 or m_1, which he outputs for A as e_A in the unforgeability game.

Since A can can perform successful forgeries, he can output such e_B that the reader will output $|set_A \cap set_B|$ bigger than the number of shared keys between two tags. For this e_B needs to be decrypted using sk into a message $c_B||set_B$. This message set_B should contain at least one value equal to a value of either set_{ran} or set_A, depending on the plaintext of e_A. Now, A' submits e_B to the decryption oracle within the IND-CCA2 game. As a result he gets the plaintext of $c_B||set_B$. Now, A' has to check if set_B contains at least one value equal to set_A, or set_{ran}. This helps him to learn if he received earlier an encryption of m_0 or m_1. Thus, A' is able to win the IND-CCA2 game.

□

Theorem 7. *If the encryption scheme $pkENC$ is IND-CCA2 secure, then $Adv^{link}_{S,A}(\lambda)$ of a (non-minimal) NARROW-STRONG adversary, that runs in polynomial time, to win the unlinkability game is negligible in the random oracle model.*

Proof. Since the protocol is symmetrical, we consider the protocol from perspective of tag Alice without loss of generality. The goal of the proof is to show that if an adversary A can break unlinkability of the protocol, the adversary A' can win the IND-CCA2 game. The adversary A' interacts with the IND-CCA2 game challenger, possessing a key pair (pk, sk). The adversary A' simulates the unlinkability game for an adversary A by answering all requests A makes to oracles. The reader key pair in the simulation is the key pair of the IND-CCA2 game challenger.

During the learning phase A' answers all the oracle queries from A' using his knowledge of tags' keys and the value of pk. To get decryptions of the messages encrypted with pk, A' queries the IND-CCA2 oracle. During the challenge phase A'

creates messages for both challenge tags $c_A||set_{A0}$ and $c_B||set_{A1}$, where set_{A0} and set_{A0} are generated according to the protocol. A' then submits both messages to the IND-CCA2 challenger. The received ciphertext $E_{pk}(c_b||set_{Ab})$ is forwarded to A. If A can break unlinkability of the protocol, thats is distinguish between tags, A' will be able to distinguish which message was encrypted.

A NARROW-STRONG adversary cannot get the result of the protocol run, but he can corrupt tags. Corrupting tags will provide an adversary with secret keys of tags. However, it does not help him in distinguishing tags and their output. This holds for the simple reason, all the information related to tags is transferred encrypted using the asymmetric IND-CCA2 encryption scheme Enc. This implies, encrypted messages do not provide an adversary with any useful information.

□

Theorem 8. *If Fun is a PRF, then $Adv^{link}_{S,A}(\lambda)$ of a polynomial-time adversary that possesses the private key of a valid reader to win the minimal unlinkability game is negligible in the random oracle model.*

Since the protocol is symmetrical, we consider the protocol from the perspective of tag Alice without loss of generality. We are going to show that an attacker is unable to distinguish any of the challenge tags from a simulator that is returning random values, and, thus, is unable to distinguish challenge tags themselves.

We now simulate the $SendTag$ oracle to an adversary as follows:

1. Upon getting initialization request, a fresh random value c_A is output.
2. Upon receiving c_B, a set of random values set_x is generated.
3. The value $c_A||set_x$ is encrypted with pk as e_A.

The above simulated tag is indistinguishable from a real tag. First output is generated the same way as in the real tag. The second value an attacker treats as an encrypted value. After he decrypts it, he obtains $c_A||set_x$. The set of random values set_x is indistinguishable from a set of randomly permuted PRFs and random values $\{f_B[i]|i \in \{1 \dots m\}\}$. The value c_A is a selected randomly. Finally, since the game is minimal, an attacker does not have any possibility to distinguish between the real tag and the simulator by matching them with other tags and comparing the output.

Assume the challenge bit $b = 0$. Tag t_0 is indistinguishable from a simulated tag. The same argument applies to challenge bit $b = 1$. Hence, an attacker cannot distinguish between two tags.

4.6 Related work

In 2012, Elkhiyaoui, Blass, and Molva presented a protocol that allows an RFID reader to determine whether two tags store some attributes that jointly fulfill a boolean constraint, without violating the privacy of the tags [123]. They motivate their protocol by considering the transportation of chemicals where safety regulations prohibit the joint transportation of chemicals that might react with each other. By equipping each container with a tag and scanning for certain boolean constraint describing reactive combinations the reader can check if the transportation is safe. Elkhiyaoui *et al.* also focus extensively on privacy of their protocol, although this is problematic for their specific application: legal regulations for the transport of dangerous goods require a clear labeling which voids any of the privacy a tag might offer [125].

The speed-dating protocols described in this chapter achieve the same goal with a different trade-offs between privacy and efficiency. Tags in our protocol are more costly, since they require to be able to perform calculations. The cost of calculations on tags is fairly low when symmetric encryption is used. Asymmetric encryption on tags is more expensive, but feasible to be implemented in both secure and efficient way, as numerous studies demonstrate in theory and practice [141, 208, 63, 163]. As observed by other researches, asymmetric primitives will provide more secure systems [311, 106]. Currently, many studies proposed protocols for which asymmetric encryption schemes are essential, [8, 106, 67] to mention a few. One of these protocols named "Yoking-Proofs" [187] is similar to speed-dating in the sense that it also considers two simultaneously scanned tags. What is different is the goal of the protocol: they provide a prove of the fact that two particular tags have been scanned simultaneously.

The advantage of using more costly tags is that the infrastructure of readers for our speed-dating protocols is more flexible and robust, because readers do not have to be connected to a central database. Additionally, unlike in the T-Match protocol presented in [123], readers do not need to perform any homomorphic encryption operations, or expensive multi-party computations.

The protocol described in Section 4.5.2 furthermore extends the protocol model to allow multiple attributes per tag. This makes speed-dating a suitable solution for a broader set of applications.

4.7 Conclusion

Three protocols to privately match attributes on RFID tags were presented in this chapter. None of them requires a centralized system with readers constantly connected to a central database. Neither do readers require any knowledge about attributes stored on tags. This makes the system flexible and easy to use for several parties. The first protocol protects privacy of users by only utilizing symmetric encryption, which makes it extremely lightweight. This comes at a cost of a slightly lower protection level, than the one provided by the other two protocols. Just one step of asymmetric encryption that is required by both of them (given hybrid encryption is used), quite noticeably changes anonymity protection.

There is a restriction in all of the presented protocols. All possible applications are limited to the ones, which are sensitive to false positives. That is, the protocol protects against matching tags with no matching attributes. These applications should not be sensitive to false negatives. The interesting future work is to see how protocol can be improved to add detection of false negatives.

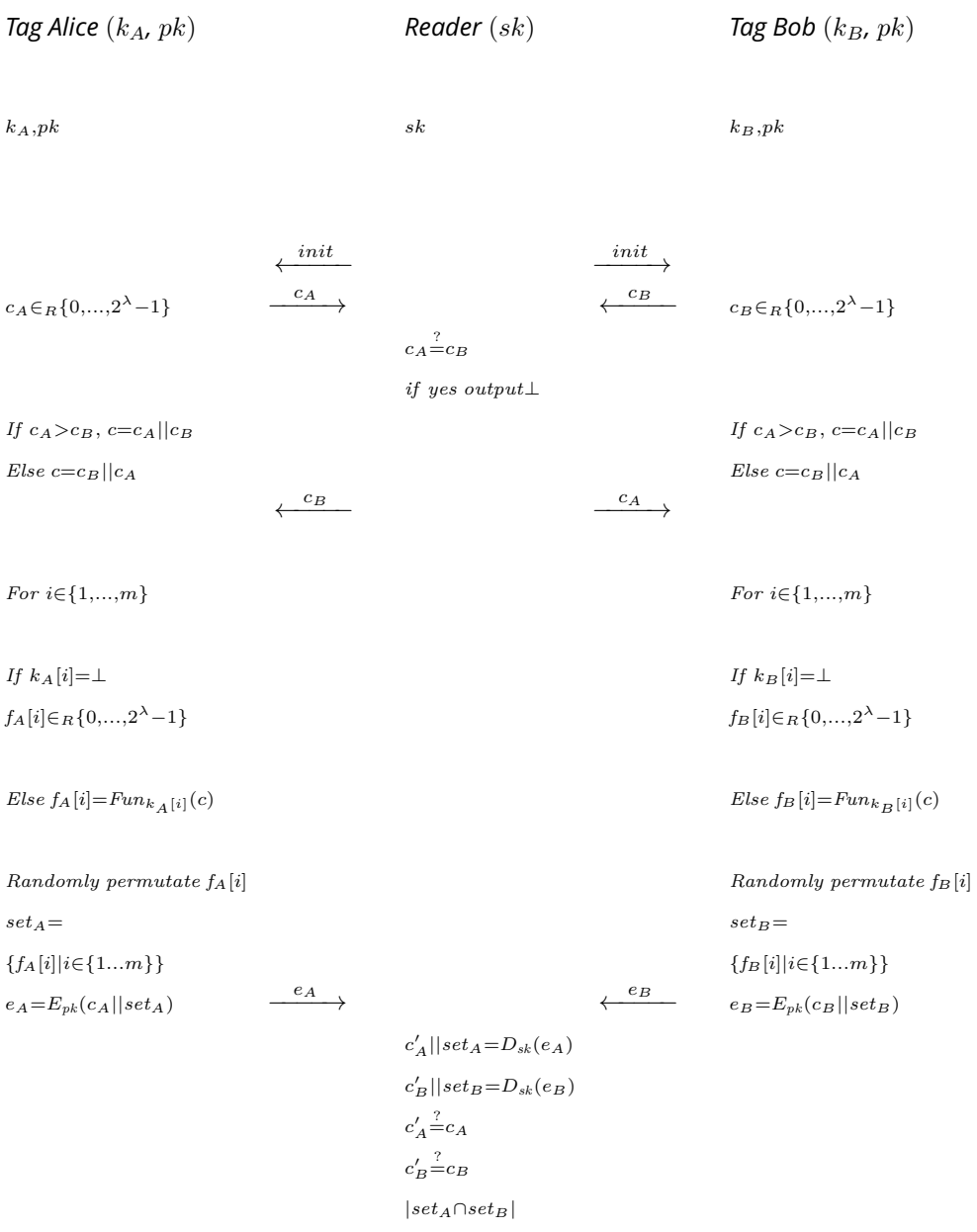

Figure 4.3: Many-keys asymmetric speed dating protocol.

Chapter 5

Footprint scheduling for Dining-Cryptographer networks

5.1 Introduction

"We kill people based on metadata" – this statement by former CIA and NSA director Michael Hayden demonstrates more than clearly that cryptographically protecting only the content of communication is insufficient; secure communication also has to protect the identities of the communicating parties. Various protocols and techniques have been proposed to enable anonymous communication. All of these techniques have to choose a trade-off between efficiency in terms of throughput, latency, and scalability on the one hand and security guarantees on the other hand. For example, anonymous proxies provide low latency and potentially very good throughput and scalability, but all participants' identities are compromised if one trusted node, the proxy, is compromised. Stronger guarantees are offered by a cascade of proxies in onion-routing networks like Tor [303], which were originally proposed by Syverson, Goldschlag, and Reed in [269]. However, onion-routing networks are susceptible to attacks that correlate traffic entering and leaving the network. See, for example, [232]. Mix-nets, proposed already in 1981 by Chaum in [80] do not have this problem; however, they suffer from increased latency.

The anonymity protocol that offers the strongest guarantees is the *Dining-Cryptographers protocol*, also known as Dining-Cryptographers network or short DC-net, which was introduced by Chaum in [83]. Interestingly, DC are also initials of the author of this protocol, as Andreas Pfitzman pointed out [259]. The protocol provides unconditional communication anonymity for senders and recipients. Additionally, it protects against data retention and anonymity "revocation": even if a central server is used and it is malicious, this server cannot reveal any information about messages and communication partners of participants. However, protection comes at the cost of low throughput and high latency, in particular when scaling to many participants.

Notation. We write 0^B for the string that consists of B zeros. All logarithms in this chapter are to the base 2.

Organization of this chapter. Section 5.2 reviews the state of the art in scheduling for DC-net. Section 5.3 introduces footprint scheduling and Section 5.4 describes how to tune the parameters of footprint scheduling and compares its performance to previous approaches. Section 5.5 describes a protection mechanism against disruption. Section 5.6 summarizes the key strengths of footprint scheduling.

5.1.1 Our contributions

This paper presents a novel approach that belongs to the class of reservation-map methods. The general problem of this class of methods is that they essentially defer the problem of undetected collisions from the communication phase to a slot-reservation phase. One way to solve this problem is to use many more slots than participants, to keep the probability of collisions low. However, this leaves many communication slots unused and drastically reduces the throughput of the DC-net. We present *footprint scheduling* as a simple and efficient way to implement the slot-reservation phase without the loss of throughput. Footprint scheduling is the first scheduling algorithm to combine reasonable communication overhead that scales logarithmic in the number of participants, absence of computation overhead, and naturally handling participants joining and leaving the network.

5.2 Existing scheduling methods

In this section we describe the previous approaches to scheduling in DC-net. As listed in the previous section, we group the algorithms into three categories: reservation maps, collision resolution, and secure multi-party computation (MPC).

5.2.1 Overview of methods

Reservation maps. Reservation maps were already introduced as one possible way to handle scheduling in the original DC-net chapter by Chaum [83]. The idea is to perform a separate scheduling phase to assign rounds to particular participants to avoid collisions. During this phase participants can reserve a round of DC-net by setting a bit of a *scheduling message* at the position corresponding to the round number. Note that also scheduling messages are sent through the anonymous DC-net channel; i.e., they are xored with all the shared keys of the other participants. Disruptions of the scheduling message can be detected with a certain probability by checking if the Hamming weight of the message is smaller or equal than the number of participants. The downside of this approach is that, because of the birthday paradox, the number of bits in the scheduling message must be quadratic in the number of participants to avoid collisions with high probability. Reservation maps are used by Herbivore, an implementation of DC-net presented by Goel, Robson, Polte, and Sirer in [148]. Herbivore optimizes the size of the scheduling message by allowing some collisions during message cycle depending on the message size (collisions for smaller messages are more likely). For performance comparison in Section 5.4 we also performed such an optimization, however we decided for optimization that does not depend on the size of the message (See Section 5.2.3 for details).

The length of the scheduling message can be reduced if instead of bits representing rounds, one would use elements of the additive group of integers modulo m [257]. After all the scheduling messages of participants are added up, elements of value 0 indicate an unreserved round, elements of value 1 indicate a reserved round, all other values indicate collisions.

Collision resolution. The second approach proposed by Chaum in [83] is to use a contention algorithm with discrete time slots and resolve collisions by retransmitting the messages. A common example of such an algorithm is slotted ALOHA [275]. In Slotted ALOHA participants pick a time slot for transmission at will. Whenever a

collision happens, participants wait for a random amount of time before they pick a new time slot for retransmission. The simplicity of the protocol is countered by the limitation of the transmission capacity of the network due to collisions.

One way to improve transmission capacity in collision resolution is through a technique called *superposed receiving*. The idea is to derive the retransmission schedule for collided messages from the result of a collision. New transmissions have to wait until a collision has been resolved. In [259, Sec. 3.1.2.3.2], A. Pfitzmann presents such an algorithm, which is an improvement of the tree algorithm that was independently proposed by Capetanakis in [78] and by Tsybakov and Mikhailov in [309]. Pfitzmann calls this algorithm *tree-like collision resolution with superposed receiving*; in the following, we refer to this algorithm as *Pfitzmann's algorithm*. Let the number of messages that collided be s, then the protocol guarantees that the collision will be resolved in exactly s retransmission steps. Additionally, this protocol guarantees fair usage of the channel for all sending participants. Note that this algorithm requires DC-net to be modified to work on integers modulo $m > 2$ instead of simple xors. This makes it possible to compute the "average" of the collided messages (treated as an integer). Participants that sent a message smaller than the average retransmit; participants that sent a larger message wait. As soon as only two messages collided, only one participant retransmits; the other message is recomputed locally. A more detailed description of the algorithm can be found in [316] and [138, Sec 3.2.2].

Another algorithm, using a similar approach, was presented in [58] by Bos and Boer. It also computes a retransmission schedule for collided messages and requires s retransmissions after s messages collided. However, it has a larger overhead in header messages and is computationally more expensive than Pfitzmann's algorithm.

Note that these superposed-receiving techniques can also be applied to slot reservation as proposed by Waidner in [316]. Pfitzmann's algorithm is then used to resolve collisions of reservation messages. Each reservation message contains the number of the round in which a participant wants to send. With this approach the traffic load does not depend on the length of the messages transmitted through DC-net as in the original Pfitzmann's algorithm.

The first protective measure against disruption during the scheduling phase was presented by Waidner and B. Pfitzmann in [317, 318]. The idea is to investigate the reservation phase in case of impossible results of the specific scheduling algorithm used. For example, in Pfitzmann's algorithm, the number of initially collided mes-

sages should be not more than the predefined maximum. To enable investigation, all the outputs during the scheduling phase are protected by output commitments. In the special phase called *palaver phase* any participant can start an investigation of the scheduling phase in order to detect disrupters.

Secure multi-party computation. In [156], Golle and Juels propose two new versions of DC-net that allow detection and identification of disrupters with high probability by using zero-knowledge proofs. They do not consider the problem of collisions (and thus reservation of rounds) in their solution; they comment that "the problem can be avoided through techniques like secure multi-party computation of a secretly distributed permutation of slots among players, but this is impractical".

Studholme and Blake propose in [294] a way to implement such a multi-party computation called *secret shuffle* by organizing a Mix-net with participants (e.g. of DC-net) serving as nodes. Encrypted round-reservation requests are transmitted through this Mix-net to obtain a secretly permuted vector of re-encrypted requests. Re-encryption is performed such that a participant can recognize his own request only after the permutation is completed. His reserved round number can be derived from the position of the request in the vector.

This idea is used in the Master's thesis of Franck [138], in which he derives a fully verifiable variant of DC-net. Later, verifiable DC-net was rediscovered in [94] and implemented under the name Verdict. The advantage of Verdict is that it allows switching between traditional DC-net and verifiable DC-net, depending on the presence of disruption. For scheduling, Verdict uses a similar approach as [294, 138] and the same as in another implementation of DC-net by the same group, Dissent [93, 324].

In [139], Franck proposes a scheduling for DC-net based on the collision-resolution protocol SICTA. This scheduling protocol is very similar to Pfitzmann's collision resolution algorithm, the only difference being that it operates with multiplication of ciphertexts instead of addition. The author proposes to use this algorithm to produce a secret shuffle of public keys of participants to establish a schedule. The protocol is non-deterministic; it achieves a maximum stable throughput (MST) of 0.924 messages per round. Disruption in the scheduling phase in this protocol is prevented by using zero-knowledge proofs.

5.2.2 Pfitzmann's algorithm

We describe tree-like collision resolution with superposed receiving algorithm in more detail. The whole algorithm is performed though superposed sending of DC-net to ensure anonymity. This superposed sending is using additive group of $\mathbf{Z}_m$, the integers modulo $m > 2$. Reservation phase consists of the number of transmission steps equal to the number of participants N. A reservation message sent by each participant consists of two values. The first value is a number, which represents a sequential number of a slot a participant is trying to reserve. The second value is a one, which is used to count number of simultaneously sent and thus collided messages. To ensure that there is no overflow, ideally $m > Nm'$, where m' is the maximum number participants can choose as a slot number.

We will informally introduce the protocol by describing steps in this figure. During the first step of reservation all participants send their messages, superposed sending results in a collision of all of them. After this an average $\lfloor \varnothing \rfloor$ is calculated. In the next step only those participants resend their messages, whose slot number is less or equal to the resulted average. Thus, the whole set is divided into two groups - the one that resends and the one that does not. On the figure resending participants are 3, 4 and 5.

Same procedure repeats and in the third step only one participant sends his message. Now only two messages are left to resolve. However, instead of making these two messages to be resend, all participants locally perform step 4. For this, they subtract from the global sum of step 2, the global sum of step 3 and compute $\lfloor \varnothing \rfloor = 4$. Now, only a participant with slot number greater or equal to 4 resends in step 5. In case both participants sent again or nobody sends, both of them chose the same slot number and none of them is thus successful in reservation. In the figure however, participant 3 resends. In step 6 participants locally calculate the slot number of the not resend message, by subtracting from the global sum of step 4 global sum of step 5.

Now the stations whose slot numbers were less than average in the first step resolve their collision. In step 7 deduce collision of their messages using information from the previous steps and calculate average. Steps 8 and 9 happen in the similar way as previous steps. In result, one needed 9 steps, out which only 5 steps included transmission of messages.

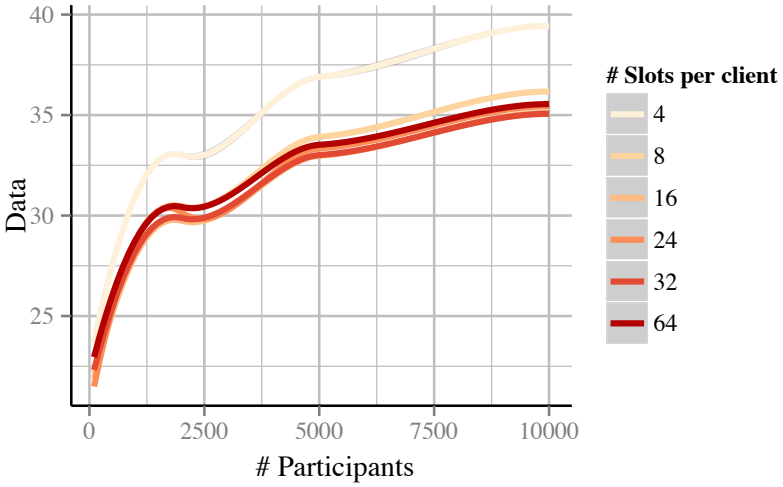

Figure 5.1: The average scheduling overhead produced by Pfitzmann's algorithm for different numbers of slots per participant.

5.2.3 Optimization of Pfitzmann's algorithm and Chaum's reservation map

When Chaum's or Pfitzmann's algorithm is used for scheduling, the number of available slots has to be chosen appropriate to the size of the network. This section briefly explains how we optimized the ratio between the number of participants and the number of slots.

Assuming that participants choose random slots in the schedule, the collision probability in both algorithms underlies the birthday paradox. However, collisions do not need to be avoided at all costs: Pfitzmann's scheduling algorithm is capable of detecting them, and Chaum's algorithm should tolerate some collisions in order to limit the scheduling overhead.

Therefore the number of slots does not need to be quadratic in the number of participants. In fact, our simulations showed that it is sufficient if there is a linear relationship between the number of slots and the number of participants. Let i be the ratio between the number of slots S and the number of participants P, so that $S = i \cdot P$. Figure 5.1 shows the scheduling overhead for Pfitzmann's algorithm with different values of i. Pfitzmann's algorithm achieves minimal overhead for $i = 32$ across various network sizes. Based on these results, the simulations shown in Section 5.4 were executed using $i = 32$.

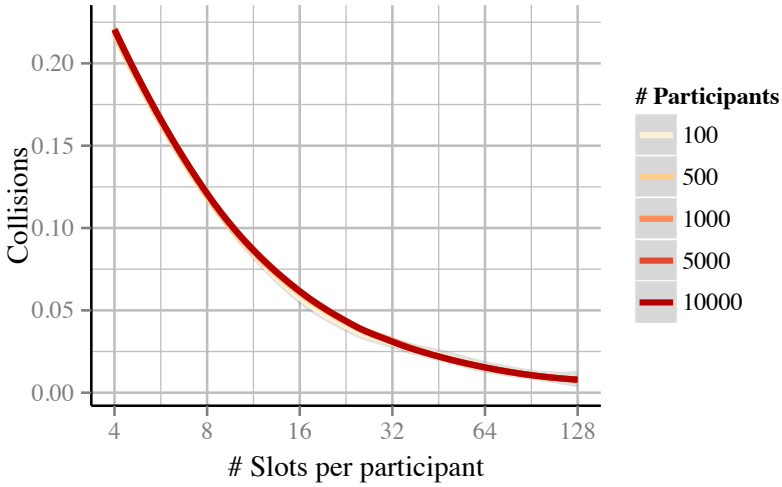

Figure 5.2: The fraction of participants that will experience a collision in Chaum's reservation map algorithm, for different ratios between the network size and the number of slots.

Chaum's algorithm cannot detect collisions and therefore needs to minimize the chance of collisions to occur. Figure 5.2 shows the fraction of participants that will experience a collision in their reservation attempt, given different values of i. We aimed for a configuration where no more than one in 20 reservation attempts would fail, which requires about 32 slots per participant.

The results for Chaum's algorithm shown in Section 5.4 were achieved with $i = 32$. If a higher success rate is desired, then the number of slots must be increased more.

5.3 Footprint scheduling

In this section, we introduce *footprint scheduling*. Footprint scheduling is similar to the map-reservation algorithm described by Chaum [83]; however, it requires significantly shorter reservation vectors and drastically decreases the likelihood of (undetected) collisions in these vectors.

In the map-reservation algorithm, the A active participants (i.e., participants who want to send a message in the next round) of a DC-net with a total of N users can reserve one out of S slots by inverting the corresponding bit in a reservation vector of S bits. The reservation vector is then transmitted through DC-net, and the resulting reservation vector, i.e., the xor of all the individual reservations, is broad-

cast to all participants. See also Section 5.2. An undetected collision occurs in this vector as soon as an odd number of participants attempts to reserve the same slot. This event is obviously undesirable, since it leads to collisions of messages during the sending phase. To decrease the probability of such an event, the original paper [83] suggests to choose the length of the reservation vector to be quadratic in the number of participants.

Footprints instead of bits. The first idea of footprint scheduling is to use $B > 1$ bits in the reservation vector to represent each slot of the schedule. The reservation vector is thus extended to a length of $B \cdot S$ bits. A participant attempts to reserve a specific slot by changing the corresponding B bits of the reservation vector to a random value $f \in \{0,1\}^B \setminus \{0\}^B$. This value is called his *footprint*. Table 5.1a demonstrates an example of a reservation vector with 3-bit footprints. DC-net will broadcast the xor of all individual reservation vectors to the participants, just as for plain map reservation. If the reservation vector contains the footprint of a participant, it is likely that no other participant tried to reserve that slot. If instead the participant finds a different value, this indicates that at least one other participant tried to reserve the same slot. In Table 5.1a one can see that participants `C`, `D` and `F` try to reserve the same slot. All three of them can recognize the collision since their original footprint s are not in the result of this round.

Table 5.1: The result of two scheduling rounds in footprint scheduling.

(a) One scheduling round of footprint scheduling with 3-bit footprints.

A	000	000	000	000	000	000	110	000
B	000	000	000	101	000	000	000	000
C	000	011	000	000	000	000	000	000
D	000	100	000	000	000	000	000	000
E	000	000	000	000	001	000	000	000
F	000	010	000	000	000	000	000	000
R	**000**	**101**	**000**	**101**	**001**	**000**	**110**	**000**

(b) The state of the reservation vector after the second scheduling round.

A	000	000	000	000	000	000	011	000
B	000	000	000	101	000	000	000	000
C	000	100	000	000	000	000	000	000
D	101	000	000	000	000	000	000	000
E	000	000	000	000	010	000	000	000
F	000	000	000	000	000	000	000	011
R	**101**	**100**	**000**	**101**	**010**	**000**	**011**	**011**

Scheduling cycles and message cycles. Using B bits per slot in the reservation vector alone would simply blow up the reservation vector by a factor of B. This is where footprint scheduling applies a second modification to reservation maps, which allows to drastically reduce the size of S (for example, to $S = 32$ for up to $10,000$ participants). The idea is to iterate slot reservation through a *scheduling cycle* consisting of R *scheduling rounds*. In the first round, each participant just attempts to reserve a slot as described above. In each subsequent round, the behavior depends on whether the participant detected a collision in "his" slot in the previous

round. If not, he will reserve the same slot again with a fresh random footprint. If the participant detected a collision, he flips a coin to choose between two possible actions. If the coin is 1, the participant backs off and does not continue to attempt to reserve any slot during this scheduling cycle. If the coin is 0, he tosses another coin[1]. If that second coin is 1, he stays in his slot and reserves it again with a fresh random footprint. Otherwise he randomly picks one of the slots that were left empty in the previous round (i.e., the ones that produced a zero xor of all footprints) and places a fresh random footprint in the corresponding slot. If no such empty slot exists, he backs off for the rest of the scheduling cycle.

In the last round of a cycle, all participants that detected a collision in their slot in the second but last round, back off and do not attempt to reserve a slot in the last round. This leaves the corresponding slots empty in the very last round. When the schedule is then executed, all empty slots can be skipped as in plain reservation maps.

Let us return to the example. After the first round, participants `A`, `B` and `E` have successfully reserved slots 7, 4 and 5, respectively. Participants `C`, `D` and `F` know that their reservation collided with reservation attempts by other participants. Slots 1, 3, 5 and 8 appear empty after the first round. Table 5.1b demonstrates the reservation vector after the second round. Participants `D` and `F` have moved away from slot 2 to one of the empty slots, while participant `C` stayed in the first slot. Note that participants `A`, `B` and `E` placed a fresh footprint in the slots they successfully reserved during the first round. They will continue to generate new footprints each round to reveal undetected collisions in case they occurred in the previous rounds.

By the end of the scheduling cycle several users hold reservations of slots in the following *message cycle*. The actual transfer of user messages in DC-net happens during this cycle. A message cycle has a maximum of S rounds, the maximum amount of slots users could reserve during the scheduling cycle.

Combining scheduling cycles and message cycles. The short length of a scheduling vector is advantageous since scheduling cycles and message cycles can now be combined to reduce latency in DC-net. For this, one has to have the number of scheduling rounds R be equal to (or smaller than) the number of slots S in the scheduling vector. Then the scheduling vector can be attached as a header to a

[1] Note that tweaking the bias of these coin tosses is necessary to reach peak performance in large networks. The pseudocode description of the algorithm in Section 5.3 shows optimal probabilities for cases where users are allowed to reserve multiple slots.

message in the message cycle to agree on the schedule of the upcoming message cycle.

Multiple reservations. The activity rate of the network participants (i.e. the percentage of users who want to send data) will depend on the application for which a DC-net is used; for an anonymous file sharing application, the activity rate might hit 100% regularly, while a chat application might have a much lower activity rate on average. The algorithm that we described up to this point is not well-suited for networks with a very low activity rate. If there are less than S active participants, and each of them is allowed to reserve exactly one slot, then the remaining slots will be unused. This has two disadvantages: On the one hand, it limits the potential throughput of small, inactive networks. On the other hand, this leaks information about the number of actively sending participants in the network. If only 4 out of 16 slots are reserved at the end of a scheduling cycle, then it is very likely that there are exactly 4 actively sending participants.

Both disadvantages can be solved by allowing all participants to reserve up to S slots. Thus, at the beginning of a scheduling cycle, each participant picks up to S slots at random, and tries to individually reserve each of them, just as described above. It is important to note that different footprints have to be used for each slot.

Pseudocode description of footprint scheduling.

Pseudocode description of footprint scheduling in Alg. 5.1 and 5.2 shows the additional steps that have to be taken in order to allow participants to reserve multiple slots.

Algorithm 5.2 Procedure for a slot-reservation attempt in footprint scheduling.

procedure SLOTRESERVE(D, F)
  $V_A \leftarrow \{\{0\}^B\}^s$
  for i from 0 to $s-1$ **do**
    if $D[i] = 1$ **then** $V_A[i] \leftarrow F[i]$
  Broadcast V_A through DC-net
  Receive V (xor of all individual scheduling vectors) from DC-net

  return V

Availability of results. To maximize re-usability of our results, we made the software used to produce simulation results publicly available at https://github.com/25A0/DCnet-simulator.

5.4 Benchmarks and comparison

This section shows how to optimize the configuration of footprint scheduling, and compares its performance to the performance of other scheduling algorithms. In the first part of this section, we will very briefly inspect the performance of footprint scheduling for different configurations in order to find optimal parameters. After that, we will compare its performance to the performance of Pfitzmann's scheduling algorithm and to Chaum's map-reservation scheduling algorithm.

Choice of parameters. There are three parameters that can be tweaked to minimize scheduling overhead: B, the number of bits per slot, S, the number of slots, and R, the number of scheduling rounds per scheduling cycle. The scheduling overhead is measured in terms of the amount of scheduling data that each participant has to send for each successful reservation that the network achieves. During one scheduling cycle, each participant will send $B \cdot S \cdot R$ bits of scheduling data. At the end of the cycle, there will be $\hat{S}$ successful reservations. Ideally, all S slots were successfully reserved, so that $\hat{S} = S$. But $\hat{S}$ might also be smaller than S if there were undiscovered collisions or unused slots in the schedule. Thus, the overhead O can be measured by

$$O = \frac{S \cdot R \cdot B}{\hat{S}} . \tag{5.1}$$

Our optimization is mostly based on this formula, and we use simulations to test how different configurations perform. Schedule convergence is an important metric for this optimization. A messaging *slot* has converged if there is at most one participant who tries to reserve this slot. A *schedule* has converged if all S slots have converged. If a slot did not converge by the end of a scheduling cycle, then no participant can successfully send a message in that slot once the schedule is executed. It is thus crucial for the throughput of the algorithm that most of the slots converge.

We will start by minimizing $R \cdot B$. Figure 5.3 shows how many scheduling rounds are necessary to reach schedule convergence[2] for different values of B. Increasing B leads to a decrease in R, but it is easy to see that $B \cdot R$ is minimal for $B = 2$, regardless of the choice of S. Changing the network size does not affect this outcome.

In the previous simulation, the number of participants was fixed. But in order to determine an optimal value for R in general, we will now look at networks of vari-

[2] Note that, while it is easy to detect in a simulation whether all collisions have been resolved, it is not trivial to detect this in practice.

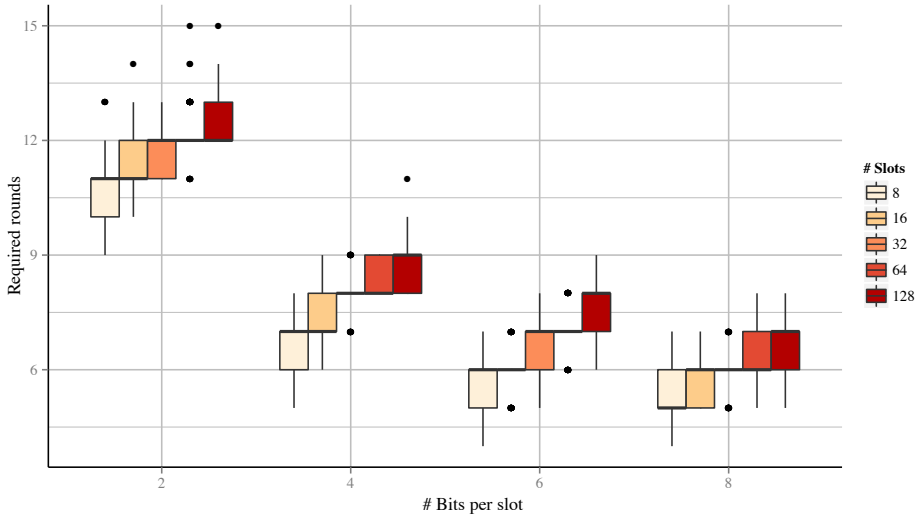

Figure 5.3: The number of rounds that are necessary to resolve all collisions, for 5000 participants and different values of B and S.

ous sizes. Figure 5.4 shows the number of scheduling rounds that are necessary to resolve all collisions for $B = 2$ bits and various values of S. The number of required rounds is clearly influenced by both, the network size and the number of slots. Although schedules converge faster on average when the number of slots is small, it is not recommended to choose $S < R$: Similar to Pfitzmann's scheduling algorithm, each message in footprint scheduling depends on the content of the previous message. Therefore, the scheduling data needs to be sent in individual packages. For $B = 2, S = 16$, these packages will only hold 32 bits of data. When this data is sent over the Internet, then the TCP-IP header of at least 32 Byte will add a massive overhead. But if $S \geq R$, then the current message cycle and the scheduling cycle for the following schedule can be interleaved as described in Section 5.3. This will decrease the *relative* overhead of the TCP-IP header and network delay. For this reason, we choose $S = 32$, which requires far less than 32 scheduling rounds, so that scheduling and message data can be interleaved.

Next, we will inspect the relation between the network size and the number of scheduling rounds. The dashed blue line in Figure 5.4 shows $\log(N)$, with N being the number of participants in the network. For $S = 32$, no more than $\log(N)$ rounds are necessary on average for the schedule to converge. Thus, rather than having one fixed value for R, we can dynamically determine R based on the current size of the network.

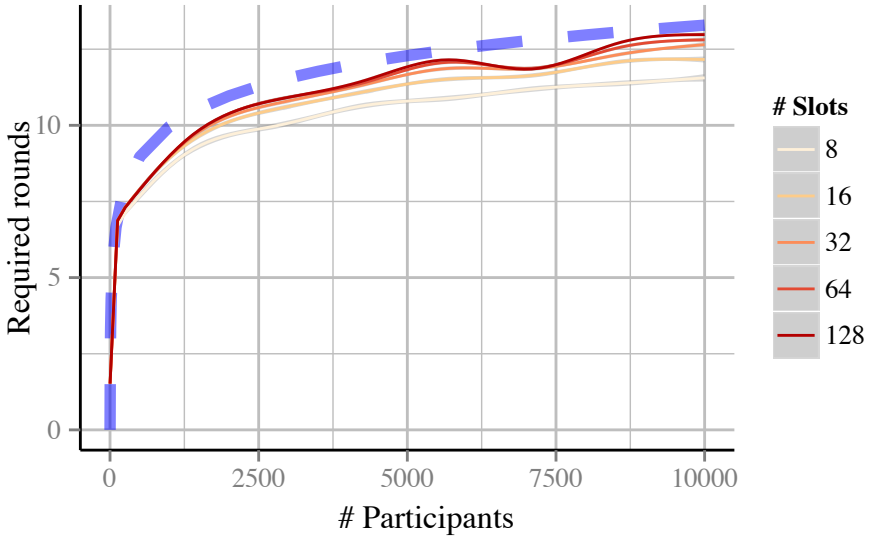

Figure 5.4: The number of rounds that are necessary to resolve all collisions for various numbers of slots and networks of different sizes. These results were produced with 2 bit footprints.

Theoretical comparison.

We provide a comparison Table 5.2 that compares 3 main scheduling methods discussed in this chapter with footprint scheduling. Let $\tilde{J}$ be an amount of bits the ciphertext of length J increases after public-key encryptions. Some explanations of the compared properties:

- *Guaranteed sending* - indicates if a user is guaranteed to have a slot reserved after one run of scheduling protocol with no adversaries present.

- *Equal throughput* - indicates if all users are assigned the same amount of reserved slots or not. Note, that Pfitzmann's algorithm used for collision resolution provides equal throughput for all sending users. However, as a reservation method it does not due to possibility of several users choosing the same slot number to reserve.

- *Adopts to load change* - indicates if the protocol can adopt to changing number of active users between two protocol runs to avoid too many empty slots or too many unsuccessful reservation attempts.

- *Disconnected users* - shows if the protocol can handle users disconnecting during run of the scheduling protocol. For example, Pfitzmann's algorithm would have to completely restart after a user is dropped out.

- *Collisions* - describes how collisions during run of the scheduling protocol are handled.

- *Scheduling overhead* - demonstrates how much data a single user has to send in order to reserve a single slot. To simplify estimations, given formulas are calculated under assumption that there are no collisions and no adversaries present. In footprint scheduling a user sends BRS amount of data (over R rounds), that leads to S slot reservation in the ideal case. Thus, for a single slot reservation, a user has to send BR data. Note, that it differs from the formula (5.1) in the assumption that reservation protocol leads to S successful reservations. According to our performance estimations, one can use R equal to $log(N)$. In Chaum's algorithm a user sends once N^2 amount data, leading to A slot reservations in ideal case. Thus, the single slot reservation requires $\approx \frac{N^2}{A}$. After optimization described in Section 5.2.3 the formula becomes $\approx \frac{32N}{A}$. Note, optimization comes at the cost of tolerating collisions during message cycle. In Pfitzmann's algorithm each can choose a slot with a number up to N, however one has to choose a larger modulo $m = N^2$ to avoid undetected collisions when adding up choices of all users. Thus one needs $log(N^2)$ bits to represent each slot number. To avoid choosing same slot number, one has to multiply N^2 with 32 (See Section 5.2.3). In addition to that, each schedule message is appended with $log(N)$ bits for counting the number of collided messages. For Dissent protocol we provide simplified formula that counts only large messages sent through the network.

- *Hides number of active users* - indicates if the protocol allows an external observer to estimate from the scheduling cycle how many users are going to send in the DC-net.

Algorithm 5.1 Footprint scheduling from the perspective of one participant A.

Parameters: Number of footprint bits B, number of participants N, number of slots S per message cycle

Output: A vector $D \in \{0,1\}^S$, indicating for each slot whether it can be used for sending.

$R \leftarrow \log(N)$
$D \leftarrow \{1\}^S$ ▷ Vector indicating which slots can be used for sending
$f \leftarrow \{0,1\}^B \setminus \{0\}^B$ ▷ Set of possible footprints
$F \leftarrow_R f^S$ ▷ Vector holding footprints for each slot
$V \leftarrow$ SLOTRESERVE(D, F) ▷ First round of the scheduling cycle
for i from 1 to $R-2$ **do** ▷ Rounds 1 to $R-2$
  for j from 0 to $s-1$ **do**
    if $D[j] = 0$ **then**
      continue
    if $V[j] \neq F[j]$ **then** ▷ Reservation attempt failed
      $c_1 \leftarrow_R [0,1)$ ▷ Biased coin toss
      if $c_1 < 0.7$ **then**
        $D[j] \leftarrow 0$ ▷ Back off
      else
        $c_2 \leftarrow_R \{0,1\}$
        if $c_2 = 1$ **then** ▷ Try same slot again
        else ▷ Empty slot available?
          $I \leftarrow \{s' | D[s'] = 0 \text{ and } V[s'] = 0\}$
          $D[j] \leftarrow 0$
          if $I \neq \emptyset$ **then** ▷ Pick empty slot
            $s' \leftarrow_R I$
            $D[s'] \leftarrow 1$
  ▷ Generate new footprints
  $F \leftarrow_R f^S$
  $V \leftarrow$ SLOTRESERVE(D, F)
for j from 0 to $s-1$ **do** ▷ Last round
  if $V[j] \neq F[j]$ **then**
    $D[j] \leftarrow 0$
$F \leftarrow_R f^S$
SLOTRESERVE(D, F)

return D

Table 5.2: Overview of main scheduling methods.

Property	Footprint	Chaum/Herbivore	Pfitzmann	Dissent
Guaranteed sending	-	-	-	+
Equal throughput	-	-	-	Guaranteed
Adopts to load change	Inherent	Estimated	Inherent	Manual configuration by group leader
Disconnected users	Compensated	Unaffected/Increases number of empty slots	Protocol fails	Protocol fails
Collisions	Resolved with high probability	Prevent reservation	Prevent reservation	Impossible
Required operations	XOR	XOR	Addition modulo $m > 2$	Public key encryption, signing, zero-knowledge proof
Scheduling overhead [3]	$B \times log(N)$, See[4]	$\approx \frac{32N}{A}$, See[5]	$log(32N^2) + log(N)$, See[6]	$\approx \frac{(A-1)(\tilde{J}+2A\tilde{J})}{A^2}$, See[7]
Amount of rounds	Adjustable, $R = log(N)$	One	A	Not applicable
Achieves privacy	Unconditional	Unconditional	Unconditional	Computational
Hides number of active users	+	-	-	-

[3] For numerical examples see Table 5.3

[4] Recommended $B = 2$, thus the formula is $2R$

[5] Optimized, See Section 5.2.3

[6] Optimized, See Section 5.2.3

[7] Simplified estimation (very optimistic)

Performance comparison. With this configuration, $S = 32, R = log(N), B = 2$, we will now show how footprint scheduling performs compared to other scheduling algorithms. For this, we will show benchmark results for three scenarios: One with a very high activity rate, as it may occur for torrent downloads and video streaming, one with a very low activity rate, which could simulate instant-messaging, as well as two scenarios with intermediate activity rates.

We will compare the performance of footprint scheduling to the performance of Chaum's reservation map, as well as to the performance of Pfitzmann's collision resolution algorithm. We optimized the ratio between the number of participants and the number of slots for both algorithms using a series of simulations. We should note that the scheduling overhead of Chaum's reservation map depends heavily on the presence or absence of an estimation of the current activity rate. An abstract description of an algorithm that can be used to predict the activity rate is given in [148, pp. 10-11]. But especially in large networks where message cycles can take multiple minutes, the number of users might change drastically between two subsequent rounds. This makes it particularly difficult to predict the activity rate of the following cycle correctly. In our simulation, we measure the performance of Chaum's algorithm for the case that there is no estimation of the activity rate. Note that footprint scheduling is only configured based on the size of the network; it does not require an estimate of the number of active users. More details on the optimization process can be found in Section 5.2.3.

Performance is measured in terms of *scheduling overhead*, as defined in the beginning of this section. Note that the size of the message itself is not taken into account in our simulations because the absolute overhead to reserve a slot is not affected by the message size. Also, while Pfitzmann's algorithm could be used to resolve collisions between actual messages, we implement it for the scheduling purposes, which is in general more efficient for reasonable message sizes.

Before presenting results of simulation, we demonstrate in Table 5.3 theoretical estimations of scheduling overhead according to the formulas given in Table 5.2 in Section 5.4. These formulas are valid in case of no collisions, which means that after completion of the scheduling protocol all slots are reserved successfully. In such conditions footprint scheduling is advantageous in all considered network sizes. However, results of simulations show somewhat different picture, which we discuss below.

In our simulations we do not consider the number of participants in the network to be larger than $10,000$. Significantly larger networks are almost always impractical

Table 5.3: Average scheduling overhead in Bytes (Based on the formulas given in Table 5.2).

	Participants						
Algorithm	100	200	500	1000	2000	5000	10,000
Footprint ($B = 2$)	13.29	15.29	17.93	19.93	21.93	24.58	26.58
Pfitzmann	24.93	27.93	31.9	34.9	37.9	41.86	44.86
Herbivore/Chaum ($A = 1\%$)	3200	3200	3200	3200	3200	3200	3200
Herbivore/Chaum ($A = 10\%$)	320	320	320	320	320	320	320
Herbivore/Chaum ($A = 50\%$)	64	64	64	64	64	64	64
Herbivore/Chaum ($A = 100\%$)	32	32	32	32	32	32	32

because the amount of message data that is produced by the entire network grows quadratically with the number of participants.

Figure 5.5 shows the performance of the three scheduling algorithms in networks with different activity rates. The scheduling overhead of both, Pfitzmann's algorithm and footprint scheduling, scale with the activity rate of the network. The overhead of Chaum's reservation maps increases with a decrease in the network's activity rate, since the available slots cannot be used as efficiently. Chaum's algorithm does, however, offer the lowest scheduling overhead for large networks. In theory, Pfitzmann's algorithm offers a scheduling overhead in large, active networks that is similar to what Chaum's reservation maps achieves, but we will address in Section 5.6 why Pfitzmann's algorithm is not practical in large networks.

In a network where the activity rate and the network size do not change drastically, the scheduling overhead can be minimized by choosing either Chaum's reservation maps or Pfitzmann's algorithm, depending on the exact characteristics of the network.

However, in a dynamic network where network size and activity rate change over time, footprint scheduling gives a better overall performance: While the activity rate is low, the network can benefit from the reduced scheduling overhead that footprint scheduling offers. In large networks and in networks with a very high activity rate, footprint scheduling still adds a slight increase in the scheduling overhead, compared to Chaum's reservation maps.

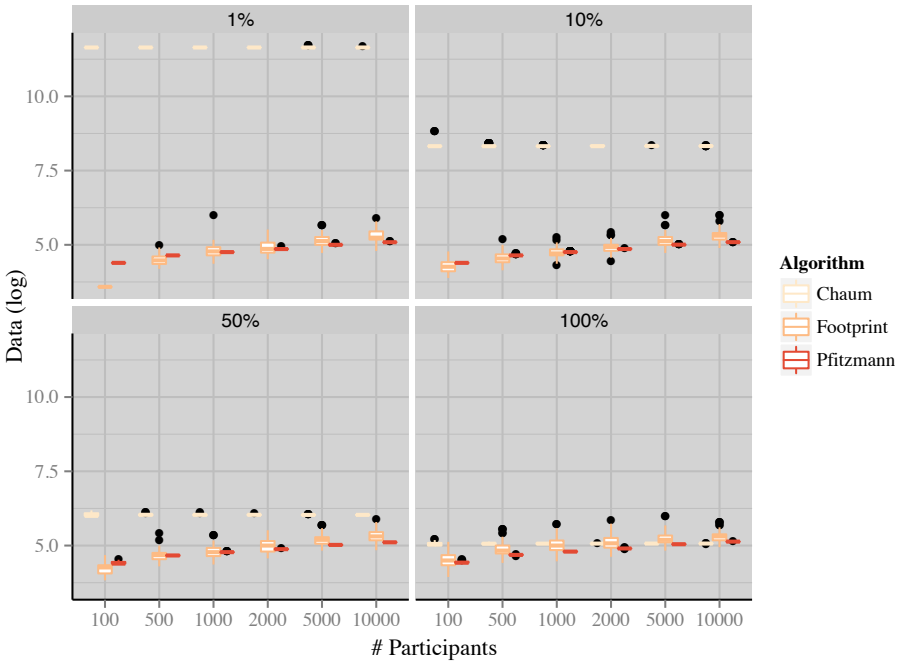

Figure 5.5: The overhead of all three scheduling algorithm in networks with different activity rates.

5.5 Disruptions and footprint scheduling

Recall that disruptions are collisions that are intentionally induced by a denial-of-service attacker or a participant who attempts to increase his transmission bandwidth on the cost of the bandwidth of other participants. In this section we briefly describe a possible protection against an attacker who attempts to disrupt the scheduling phase of footprint scheduling.

The literature describes two approaches to cope with disruption in DC-net. The first approach is to open up messages of participants after (suspected) disruption [83, 58, 318] and thus reveal which participants did not behave according to the protocol. The second approach is to use zero-knowledge proofs [156, 93, 94, 139]. Our technique follows the first approach since it does not require any of the two computationally-secure variants of DC-net introduced in [156]. Additionally, scheduling messages can be opened without compromising anonymity of participants. This holds if the opened schedule is afterwards discarded and if the sending rates are constant, so sending wishes of a particular participant cannot be learned.

The idea is to use a PRNG with a secret seed for all randomness that is required for footprint scheduling (i.e., slot positions, footprints and random choice to stay in a slot or back off). To prevent cheating, users are obliged to commit to the seed. Note that this is an obvious choice also for efficiency reasons. To protect against disrupters, each participant uses a new random seed for every scheduling cycle and commits to this seed before scheduling.

Whenever the decision is made to open a scheduling cycle, each participant publishes the seed used for this cycle. These seeds are checked against the commitments and then the scheduling vectors of each round are recomputed and compared with the scheduling vectors that were previously obtained from the DC-net output. Note, keys and messages output by each individual participant are not opened and verified at this stage. If these recomputed scheduling vectors match, all participants followed the rules. If not, at least one of the participants did not follow the protocol. In order to find the disrupter, all participants reveal their keys used in the scheduling phase. To prevent disrupters from wrongly accusing honest participants by revealing an incorrect key, one can enforce that participants also commit to those shared keys in advance.

Such a technique provides performance improvements when the scheduling algorithm has a certain chance of undetected collisions (Footprint, Herbivore) for the following reason. Undetected collisions during the scheduling cycle lead to collisions during the message cycle. How can one efficiently distinguish an honest colli-

sion of messages due to such a problem and a disruption? Traps (non-meaningful messages that can be opened with no harm to anonymity) will not be helpful to answer this question for every single case of messages colliding. Opening of keys and rounds is too costly for such a check, ideally one would want to apply heavy methods only if it is known that there is a disrupter. Our method allows to perform a quick and efficient check if there was a disruption or not by opening only seeds used for generating footprints, and only after that decide if to open the scheduling cycle, which involves opening keys shared between users and verifying if individual outputs were made correctly.

5.6 Conclusion. Advantages of footprint scheduling

In this section, we provide a detailed description of the main advantages of the footprint algorithm.

As mentioned earlier, footprint scheduling inherits from the reservation-map algorithm. In particular, footprint scheduling involves no computational overhead for participants. The algorithm improves on reservation maps by reducing the probability of undetected collisions in the reservation vector. In networks with very high activity rate the cost for this improvement can be a very slight increase in scheduling overhead, depending on how many undetected collisions the reservation-map algorithm accepts. If message collisions are prohibitive or if the network does not have a very high activity rate, footprint scheduling noticeably *reduces* the scheduling overhead compared to reservation maps. For details see Section 5.4.

Further, unlike superposed receiving and MPC-based scheduling protocols, footprint scheduling naturally handles events of participants joining or leaving the DC-net during the schedule negotiation. When a participant disconnects, his reservation slot will appear free in the next round. Any participant that is in the process of resolving a collision can now move to this slot. Thus, footprint scheduling reallocates slots that become available, even in the middle of a schedule cycle. At the same time footprint keeps scheduling and message cycles short, which permits fast joining to the network. That in turn improves anonymity, allowing potentially a bigger anonymity set in the new cycle.

When using Chaum's reservation map, a good estimate of the network's activity rate is necessary in order to optimize the scheduling overhead (for details, see Section 5.4). With footprint scheduling, the success chance of a reservation attempt automatically increases if fewer participants bid for a slot in the next message cy-

cle. Senders will be able to reserve a slot in fewer attempts if the activity of other participants goes down, and it will take more attempts if other participants become more active. This makes it unnecessary to estimate the activity rate.

Just like Pfitzmann's scheduling algorithm, footprint scheduling is an interactive protocol, in the sense that each message in the protocol depends on the content of the previous one. For Pfitzmann's algorithm, the protocol is completed after A successive messages, where A is the number of active network participants. In the case of footprint scheduling, the protocol is completed after S successive messages, where $S \ll A$ for large networks, as we showed in Section 5.4. In practice, network latency alone can be a major obstacle to complete a protocol of A messages in a large network. The following example shows a best-case scenario for a network with $10,000$ active participants: Assume that all participants live in major US cities. In this case, the average latency between them will be about $33ms$ on average at the time of this writing[8]. Thus, there will be a delay of at least $66ms$ between each message. With $10,000$ active participants, this means that the protocol is completed after $10,000 \cdot 66ms = 660s = 11min$.

Recall that Pfitzmann's scheduling protocol cannot be completed if a participant leaves before the protocol is completed. It is impractical to demand that not a single participant must leave the network over a period of 11 minutes, especially when some participants are connected via personal mobile devices like a phone or a laptop. Footprint scheduling does not suffer from this problem since each protocol run is completed much faster, but also since the protocol can be completed even if users disconnect from the network.

Footprint scheduling is also advantageous due to the fact that it hides the number of actively sending users in the network from an eavesdropping adversary. Such information can serve as a marker of upcoming social events; for example, the Tor network showed largely increased activity just before the Arab Spring[9].

One of the advantages of DC-net over Mix-nets (and onion routing) is that it hides the number of actively sending users due to the fact that all the users have to contribute to the network in order to facilitate anonymous sending. Unfortunately, previous efficient scheduling protocols either allow to estimate the number of active users by counting the number of empty slots in the scheduling messages, or they require to know the number of active users to operate.

Footprint scheduling disguises the number of active users as long as each user is allowed to reserve multiple slots. Even very small networks, the number of free

[8] http://ipnetwork.bgtmo.ip.att.net/pws/network_delay.html

[9] http://www.monitor.upeace.org/innerpg.cfm?id_article=816

slots does not give away the number of active users. An internal observer can gain a rough estimate of the number of active users over a longer period, based on the number of collisions that they experience. However, for an external observer, it is impossible to determine reliably whether there is a collision in any of the slots, since footprints change with every round. Further, for an external observer it is impossible to estimate if there were collisions in any of the slots since footprints change from round to round even if there were no collisions. In footprint scheduling the number of slots as well as discussion rounds does not change with the number of active users. Altogether, this prevents estimation of actively sending participants if footprint scheduling is used.

Last but not least, footprint scheduling has an advantage over other map reservation protocols due to fast and efficient method of verifying if a collision in message cycle was caused by an undetected collision in scheduling cycle or by a disruption. For details, see Section 5.5.

Chapter 6

Elligator: Elliptic-curve points indistinguishable from uniform random strings

6.1 Introduction

The motto of the German colonial empire of 19th century “Wissen ist Macht, geographisches Wissen ist Weltmacht” (Knowledge is power, geographical knowledge is world power) can be transformed to an up-to-date version: “Knowledge is power, restricting knowledge is world power”. A recent study shows, that out of 74 countries 39 perform some degree of censoring of Internet access [177]. The study is performed by the OpenNet Initiative that collected data in the period between 2007 and 2012 (years of data collection differ by country). News of the last years [323], [221], [219], [26] indicate, that the situation likely became only worse.

Censoring is not performed solely by governments. Companies and public organizations widely adopt content-control software to restrict access to information that goes beyond what is requested by government [252], [164], [307]. Especially dangerous, when it is used by search engines [238].

Censorship circumvention tools. Non-surprisingly the current situation lead to an active use of censorship circumvention tools. For example, as statistics collected by GlobalWebIndex [329] demonstrates, 1 in 4 of Virtual Private Networks (VPN) users

in 2016 access this kind of service to bypass censoring. Another popular tool is the widely used anonymous communication system called Tor[1] [113].

Censorship circumvention is the very useful "side-effect" of Tor and other anonymous communication systems. They help bypassing a censor who is maintaining a blacklist of forbidden IP addresses and denies all requests to them. When a user is sending a message to such an IP address through the anonymous communication system, the censor does not observe the IP address from the blacklist, but an IP address of one of the nodes of the anonymous communication system itself. This prevents filtering out user requests by their destination addresses.

The quality of communication freedom protection by anonymous communication systems can be judged by the fact that a large number of countries are either discussing [230] or have already decided to block usage of anonymous communication systems, in particular Tor [305].

Authorities come up with more and more ways to block popular bypassing tools, and Tor is no exception. Its users base is estimated to be larger than 2 million users each single day[2]. During social events the active number of users can significantly grow, as it happened just before the Arab Spring [330]. This starts a non-stopping arms-race: new ways to block anonymous communication are invented, new ways to bypass restrictions are presented as a response. The cycle goes on [13].

Encryption is helpful against filtering based on the keywords performed using inspection of the content of the network messages using deep-packet inspection (DPI) hardware. It prevents packets to be suppressed based on their content alone. Another approach to filtering using DPI is blocking of specific protocols: authorities distinguish characteristic patterns in the traffic and correlate it with typical patterns of undesired protocols.

In response to that scientific and activists communities come up with more and more elaborate ways to keep usage of Tor undetected (so called "pluggable transport" [306]). Pluggable transport reshapes traffic to match some allowed protocol, for example Skype. For this to work, one adds another layer of encryption (superencrypt), which itself can be detected if protocol to mimic is using different type of encryption or no encryption at all. One has to note, that encrypted traffic itself already has been a target to blocking by governments [68], [284] and organizations [307].

Elliptic-curve cryptography and censorship circumvention. Elliptic-curve cryptography (ECC) is arguably the most important tool in modern public-key cryptog-

[1] https://www.torproject.org

[2] https://metrics.torproject.org/userstats-relay-country.html

raphy. It provides public-key encryption, signatures, non-interactive key exchange, and many higher-level security features. It offers an attractive combination of high security, high speed, and (often critical for deployment) small space consumption.

However, for applications in censorship circumvention, ECC has a security problem. ECC protocols naturally send elliptic-curve points in the clear as long-term public keys, ephemeral public keys, ciphertext prefixes, challenges, etc. These points, even in compressed form, are obvious: they are easy to distinguish from uniform random strings.

There have been some ad-hoc workarounds for this problem, notably for ElGamal ciphertext prefixes, using a curve-or-twist technique introduced by Möller (see Section 6.3.2). But each new ECC-based protocol faces the same problem. Protocol designers unaware of the issue continue building protocols that are trivially visible to attackers. Designers requiring keys and ciphertexts to be indistinguishable from uniform are forced to modify those protocols, hoping that the modifications do not compromise other forms of security.

The main goal of this work is to eliminate this problem. The solution presented here works for a wide range of elliptic-curve protocols, essentially every protocol in which the transmitted curve points are generated at random. There is no longer any need for, e.g., ad-hoc handling of ciphertext prefixes; this chapter's technique works for all of the types of elliptic-curve points mentioned above.

6.2 Preliminaries

Elliptic curves (EC) were introduced to cryptography independently by Miller [227] and Koblitz [196] and since then became widespread. The main advantage gained by using ECC in public-key cryptography is the drastically smaller size of a key compared to public-key systems based on arithmetic in finite-field groups. The difference of the key sizes according to recommendations of Ecrypt II Yearly Report on Algorithms and Keysizes for 2011-2012 [291] is the following. An EC key of 256-bit size corresponds to 3248-bit RSA key, both having 128-bit security, which means that the best known attack to recover plain text will have complexity 2^{128}.

Elliptic curves in Weierstrass form. An elliptic curve $E(\mathbf{F}_q)$ defined over a finite field $\mathbf{F}_q$ is a group of points (x, y) described by an equation of a form

$$E(\mathbf{F}_q) : y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6\,, \tag{6.1}$$

where $a_1, a_2, a_3, a_4, a_6 \in \mathbf{F}_q$ and $(x, y) \in \mathbf{F}_q \times \mathbf{F}_q$, the discriminant $\Delta \neq 0$, and an additional point at infinity $\mathcal{O}$. This curve is said to be in Weierstrass form. It is *smooth*, which means it has no points at which two or more distinct tangent line exist thanks to the condition $\Delta \neq 0$. The discriminant is calculated as follows:

$$\begin{cases} \Delta = -d_2^2 d_8 - 8d_4^3 - 27d_6^2 + 9d_2 d_4 d_6 \\ d_2 = a_1^2 + 4a_2 \\ d_4 = 2a_4 + a_1 a_3 \\ d_6 = a_3^2 + 4a_6 \\ d_8 = a_1^2 a_6 + 4a_2 a_6 - a_1 a_3 a_4 + a_2 a_3^2 - a_4^2 \,. \end{cases} \tag{6.2}$$

The *short Weierstrass equation* is capable of describing all ECs defined over large prime fields [96]. In fact, this curve form is often used to define standardized curves, for example NIST curves [241] or Brainpool curves [60]. An elliptic curve defined over a field $\mathbf{F}_q$ has the following short Weierstrass equation:

$$y^2 = x^3 + ax + b\,, \tag{6.3}$$

where $a, b \in \mathbf{F}_q$ and $(x, y) \in \mathbf{F}_q \times \mathbf{F}_q$, the characteristic of the field $\mathbf{F}_q$ $char(\mathbf{F}_q) \neq 2,\ 3$ and the curve is smooth, that is, $\Delta = 4a^3 + 27b^2 \neq 0$.

The set of points on an EC together with a point at infinity i.e. $E = \{(x, y) : y^2 = x^3 + ax + b\} \cup \{\mathcal{O}\}$ has an abelian group structure with the following group law:

1. The point at infinity $\mathcal{O}$ is the identity element.

2. The inverse of a point $P = (x,\, y) \in E(\mathbf{F}_q)$ is the point $-P = (x,\, -y) \in E(\mathbf{F}_q)$.

3. Adding points $P = (x_1, y_1)$ and $Q = (x_2, y_2)$, where $P \neq \pm Q$, gives a point that is on the curve $P + Q = (x_3, y_3)$ calculated as

$$\lambda = \frac{y_1 - y_2}{x_1 - x_2}$$

$$x_3 = (\lambda)^2 - x_1 - x_2 \quad \text{and} \quad y_3 = \lambda(x_1 - x_3) - y_1\,.$$

4. Doubling a point $P = (x_1, y_1)$ gives a point that is on the curve and is calculated as

$$\lambda = \frac{3x_1^2 + a}{2y_1}$$

$$x_3 = (\lambda)^2 - 2x_1 \quad \text{and} \quad y_3 = \lambda(x_1 - x_3) - y_1\,.$$

The presented addition formula is not *complete*, because it has exceptional points, that have to be handled differently. The new complete addition formulas for prime order elliptic curves can be found in [272].

Montgomery curves. Montgomery form of elliptic curves was introduced in [229] to speed up the Pollard and Elliptic Curve Methods of integer factorization. Montgomery form of elliptic curves over a field $\mathbf{F}_q$ with $char(\mathbf{F}_q) \neq 2$ is:

$$E(\mathbf{F}_q) : by^2 = x^3 + ax^2 + x\,, \tag{6.4}$$

where $b(a^2 - 4) \neq 0$, $a,\ b \in \mathbf{F}_q$ and $(x, y) \in \mathbf{F}_q \times \mathbf{F}_q$.

The group law of a Montgomery curve is defined as follows:

1. The point at infinity $\mathcal{O}$ is the identity element.
2. The inverse of a point $P = (x,\, y) \in E(\mathbf{F}_q)$ is the point $-P = (x,\, -y) \in E(\mathbf{F}_q)$.
3. Adding points $P = (x_1, y_1)$ and $Q = (x_2, y_2)$, where $P \neq \pm Q$, gives a point that is on the curve $P + Q = (x_3, y_3)$ calculated as

$$\lambda = \frac{y_2 - y_1}{x_2 - x_1}$$

$$x_3 = b\lambda^2 - a - x_1 - x_2 \quad \text{and} \quad y_3 = \lambda(x_1 - x_3) - y_1\,.$$

.

4. Doubling a point $P = (x_1, y_1)$ gives a point that is on the curve and is calculated as

$$\lambda = \frac{3x_1^2 + 2ax_1 + 1}{2by_1}$$

$$x_3 = b\lambda^2 - a - 2x_1 \quad \text{and} \quad y_3 = \lambda(x_1 - x_3) - y_1\,.$$

.

Elliptic curves in Edwards form. The Edwards form of elliptic curve was introduced in 2007 by Edwards [122], and proposed for ECC by Bernstein and Lange [49]. This form facilitates fast implementations of curves interesting for cryptographic applications due to fewer multiplications required for their group operation than the other popular curve forms [46], [49].

The equation for an Edwards curve currently used in cryptography and presented below (see (6.5)) is due to Bernstein and Lange [49]. An Edwards curve is defined over a field $\mathbf{F}_q$ with $char(\mathbf{F}_q) \neq 2$ by the equation:

$$E(\mathbf{F}_q) : y^2 + x^2 = 1 + dx^2y^2\,, \tag{6.5}$$

where $d \in \mathbf{F}_q \backslash \{0, 1\}$, $(x, y) \in \mathbf{F}_q \times \mathbf{F}_q$.

The group law of Edwards curves is strongly *unified*, which enables resistance to some simple side-channel attacks. Strongly unified means that addition law also handles point doubling, making two operations hard to distinguish. If d is a non-square in $\mathbf{F}_q$, then the addition law is complete and the curve is called *complete Edwards curve* [49, Theorem 3.3], which means that there are no exceptional points. The group law is defined as follows:

1. The identity element is the point $(0, 1)$.
2. The inverse of a point $P = (x, y) \in E(\mathbf{F}_q)$ is the point $-P = (-x, y) \in E(\mathbf{F}_q)$.
3. Adding points $P = (x_1, y_1)$ and $Q = (x_2, y_2)$ gives a point that is on the curve $P + Q = (x_3, y_3)$ calculated as

$$P + Q = \left(\frac{x_1 y_2 + y_1 x_2}{1 + d x_1 x_2 y_1 y_2}, \frac{y_1 y_2 - x_1 x_2}{1 - d x_1 x_2 y_1 y_2}\right). \tag{6.6}$$

Group order. The group order of an elliptic curve group defined over $\mathbf{F}_q$ is the number of F_q-rational points on this curve. F_q-rational points are points with coordinates in $\mathbf{F}_q$. Further in the text they are refereed to as points for simplicity. An estimate of the number of points on an elliptic curve over a finite field $\mathbf{F}_q$ is given by the *Hasse's theorem on elliptic curves*. Let the number of points on a curve be denoted as $\#E(\mathbf{F}_q)$. Then the Hasse's theorem states:

$$\#E(\mathbf{F}_q) = q + 1 - t, \text{ where } |t| \leq 2\sqrt{q}. \tag{6.7}$$

Alternatively, the number of points on $E(\mathbf{F}_q)$ is within the interval $q + 1 - 2\sqrt{q} \leq \#E(\mathbf{F}_q) \leq q + 1 + 2\sqrt{q}$. Because $2\sqrt{q}$ is very small relative to q, $\#E(\mathbf{F}_q) \approx q$.

If $char(\mathbf{F}_q)$ divides t, then the curve is called *supersingular*, otherwise the curve is called *ordinary* or *non-supersingular*. Supersingular curves are used in pairing-based cryptography [197], otherwise one choses non-supersingular curve to prevent pairing attacks and still can maintain efficient arithmetic [105, p. 169], [92, p. 289, p. 530].

Twists. An elliptic curve E defined over $\mathbf{F}_q$ has an associated *twist* E' when E' is isomorphic to E over the algebraic closure of $\mathbf{F}_q$. If E' is isomorphic to E over a quadratic extension of $\mathbf{F}_q$, it is called a *quadratic twist*.

Quadratic twists of Weierstrass curves have an equation:

$$y^2 = x^3 + ar^2 x + br^3, \tag{6.8}$$

where $r \in \mathbf{F}_q$ is a *quadratic non-residue*. This means that there is no integer $x \in (0, q)$ such that the equation $x^2 = r (mod\, q)$ has a solution.

Quadratic twist of Montgomery curves has the following equation:

$$\frac{b}{r}y^2 = x^3 + ax^2 + x\,, \tag{6.9}$$

where $r \in \mathbf{F}_q$ is a quadratic non-residue.

Quadratic twist of Edwards curves have the following equation [45]:

$$y^2 + ax^2 = 1 + dx^2y^2\,, \tag{6.10}$$

where $a \in \mathbf{F}_q \backslash \{0\}$.

Decisional Diffie-Hellman assumption. The decisional Diffie-Hellman (DDH) problem is one of the three problems often used for proving security of EC cryptosystems, the other two are the computational Diffie-Hellman problem and gap Diffie-Hellman problem.

Let cyclic group **G** have prime order q and generator P. Let a, b and c be chosen randomly and uniformly from the set $\{1, \ldots, q-1\}$. The DDH problem requires to decide if $cP = abP$ or not. The cryptosystem is said to be DDH secure if these is no such probabilistic polynomial time algorithm $\mathcal{A}$ for **G** that for a fixed security parameter λ and sufficiently large n has the following advantage:

$$|Pr[\mathcal{A}(P, aP, bP, abP) = \text{“true”}] - Pr[\mathcal{A}(P, aP, bP, cP) = \text{“true”}]| > \frac{1}{n^\lambda}\,.$$

6.3 Defining the problem

6.3.1 Distinguishers

We use the standard NIST P-256 elliptic curve as an example to illustrate the difficulties. A public key on the NIST P-256 elliptic curve is a pair (x, y) of integers satisfying the equation $y^2 = x^3 - 3x + b$ modulo the prime $2^{256} - 2^{224} + 2^{192} + 2^{96} - 1$, where b is a standard constant. There are at least three obvious ways for an attacker to distinguish this public key from a uniform random string:

- Least severe: Normally x and y are represented as integers between 0 and $2^{256} - 2^{224} + 2^{192} + 2^{96} - 2$ inclusive, encoded as 256-bit strings. If the attacker sees a 256-bit string representing an integer larger than $2^{256} - 2^{224} + 2^{192} + 2^{96} - 2$ then the attacker knows that the string is not a valid value of x or of y.

If the attacker sees a user sending a series of (e.g.) 2^{35} 256-bit strings, and the largest string is close to $2^{256} - 2^{224}$, then the attacker is reasonably confident that the user is sending curve points. See Section 6.4.6 for details.

One can dismiss this attack as being too slow to be of interest, but we prefer more robust cryptographic primitives that maintain security with heavy use. The user can cover all integers between 0 and $2^{256} - 1$ by randomizing the representations of small integers x and y, but this adds very little security: the attacker easily collects statistics showing that some integers appear half as often as others.

A secure solution is to randomly represent x and y as integers between 0 and, e.g., $2^{320} - 1$. Another secure solution, with smaller keys, is to switch to an elliptic curve using a prime much closer to a power of 2, such as NIST P-224 (prime $2^{224} - 2^{96} + 1$, lower security level) or Bernstein's Curve25519 from [43] (prime $2^{255} - 19$).

- More severe: The attacker simply checks the curve equation. If a 512-bit string has the form (x, y) where $y^2 = x^3 - 3x + b$ modulo this prime then the attacker is confident that the user is sending a public key.

  Many ECC systems save space by compressing y to one (random-looking) bit: the sign in $\pm\sqrt{x^3 - 3x + b}$. Some ECC systems save space by eliminating y entirely. The cost for the legitimate user of computing a square root is almost always outweighed by the benefit of reducing keys to half size. Both of these mechanisms have the side effect of stopping this attack.

- Most severe: The attacker checks whether $x^3 - 3x + b$ is a square modulo this prime. This has chance $1/2$ of occurring for a uniform random string, but if it occurs repeatedly then the attacker is reasonably confident that the user is sending public keys.

The third attack is difficult to stop. Our solution requires a quite drastic change in how curve points are represented as strings; this representation is the main topic of this work.

Our solution is not limited to public keys: it also protects other randomly generated elliptic-curve points, such as the points appearing in ciphertexts in elliptic-curve versions of the ElGamal encryption system and points appearing in signature systems. See Section 6.4.

6.3.2 Previous work

Several years ago, in [228], Möller proposed a variant of the ElGamal encryption system that provides indistinguishability for ciphertexts as follows:

- Alice's public key is a pair $(aP, a'P')$ where a and a' are secret integers, P is a standard base point on an elliptic curve $E(\mathbf{F}_{2^n})$ over some binary field $\mathbf{F}_{2^n}$, and P' is a standard base point on a nontrivial quadratic twist $E'(\mathbf{F}_{2^n})$ of the same curve.
- To encrypt a message m using public key $(aP, a'P')$, Bob chooses a random integer b, chooses randomly between aP and $a'P'$, sends the x-coordinate of bP or bP' respectively, hashes $b(aP)$ or $b(a'P')$ respectively to obtain a secret key, and sends an encryption of m using this secret key.
- Alice recovers bP or bP' from the x-coordinate. Alice multiplies bP or bP' by a or a' to obtain abP or $a'bP'$, hashes it to obtain the same secret key, and decrypts the message. Möller requires Alice to perform tests to decide whether the input is on E or on E' and to recover the whole point bP or bP'. We comment that the "Montgomery ladder" handles both cases together if a' is chosen as a. (For Montgomery-ladder background see [229], [66], and, for the binary case, [217].) This eliminates the need for such tests and offers efficient curve arithmetic.

The idea in Möller's proposal is that each element of $\mathbf{F}_{2^n}$ is a valid x-coordinate on E or its twist. The x-coordinate sent here can therefore be any element of $\mathbf{F}_{2^n}$, and in fact the distribution of the x-coordinates is indistinguishable from the uniform distribution of n-bit strings. Möller slightly adjusts the choices so that the distribution is exactly uniform.

Young and Yung present in [328] a modified version of Möller's twist idea to achieve DDH security in the standard model. In their version Alice's public key consists of M points on each curve; Bob picks one of the curves and computes shared secrets with all M points on that curve. The shared key of length M is derived taking just one bit per DH shared secret.

Our approach encodes points on a *single* curve as strings indistinguishable from uniform random strings. This has several obvious advantages over Möller's system. We obtain indistinguishability not just for ciphertexts but also for public keys. Möller's system has double-length public keys, while our public keys have minimal length. Möller's approach is limited to ElGamal encryption, while our approach handles a much wider range of ECC systems; see Section 6.4. Möller needs crypto-

graphic security not just for E but also for E'; in our system *twist security* - a requirement of having a large prime dividing the number of points on the quadratic twist of the curve - is not required. We still suggest it as a desirable feature (see Section 6.6). Since our proposal does not need to work with twists, achieving solutions in the standard model can be done simply using the standard methods.

6.3.3 Application context

Möller's curve-or-twist approach is used in StegoTorus [320]. StegoTorus is an extension to Tor [113] for censorship circumvention. It makes Tor traffic resemble Skype, general HTTPS traffic, etc. and relies on secure communication with a StegoTorus server. Establishing a link with this server requires deriving temporary key material using the public key of the server. This key material is used to encrypt communication of a subsequent ephemeral DH key exchange. StegoTorus uses Möller's example parameters over $\mathbf{F}_{2^{163}}$.

For comparison, typical ECC protocols carry out a DH key exchange in the clear; see, e.g., the ntor example below. StegoTorus needs to encrypt its key exchange because the points sent in the key exchange are easily distinguishable from uniform random data. This encryption adds the outer Möller layer, doubling the space used for the initial communication. This also slows down the client, slows down the server, doubles the size of server public keys (one point on the curve, one point on the twist), and requires implementations to handle computations on two curves.

We eliminate all of these issues by solving the underlying problem, namely the point distinguishability. The points that we send in a DH key exchange are indistinguishable from uniform random strings. See Sections 6.4, 6.5, and 6.7.

Telex [326] is another censorship-circumvention tool. Telex messages pose as regular TLS messages to random uncensored sites; the only difference is that the nonce field contains a cryptographic value instead of a random value. The public key of the Telex server is a pair of points (aP, aP'), with P on an elliptic curve $E(\mathbf{F}_q)$ and P' on a nontrivial quadratic twist. Telex servers use deep-packet inspection on all traffic passing through them and identify Telex messages by checking whether the nonce field interpreted as $\alpha||\beta$ satisfies the following conditions. Interpret α as the x-coordinate of a point R on E or E' and compute aR on the appropriate curve. Identify the message as a Telex message if β matches a salted hash of aR, and route it according to some other, encrypted information.

The use of curves in Telex follows Möller's proposal except for the choice of finite field and the choice of equal secrets. The proposed parameters have $q =$

$2^{168} - 2^8 - 1$ so that the distribution of values in $\mathbf{F}_q$ is indistinguishable from that of length-168 strings. The attempted decryption performs a complete point recovery from x and uses standard Weierstrass-curve arithmetic to compute the scalar multiplications. We emphasize again that the "Montgomery ladder" is more efficient, handling both cases together. We also comment that the system could use somewhat larger finite fields, gaining security, without reducing the 56-bit hash size and without going beyond the standard TLS 224-bit nonce size: servers would allow only points R with a fixed number of implicit trailing zeros, and clients would repeatedly generate points until meeting that condition.

This chapter's solution to the point-distinguishability problem would not save bandwidth for Telex connections but would still simplify implementations, removing any need to handle the twist, and would reduce public keys to half size, potentially a useful feature for small clients keeping track of many different Telex servers.

As a third protocol we consider ntor [150], a handshake protocol proposed for Tor achieving anonymity and one-way authentication with forward secrecy. This protocol assumes that Tor relays have public keys on an elliptic curve. The following is a simplified version of the ntor protocol, skipping certificates and saved session states, but presenting all parts relevant to the choice of curve group and representation. To extend a Tor circuit to a relay with public key $B = bP$ a client picks a random value x, computes $X = xP$, and sends X to the server. The server picks a random y, computes $Y = yP$, computes two secret keys as $k'||k = H(yX, bX, B, X, Y)$ for some hash function H, computes $t_B = \mathsf{MAC}_{k'}(B, X, Y)$ (an authenticator under key k'), and sends $Y||t_B$ to the client. The client computes $\bar{k}'||\bar{k} = H(xY, xB, B, X, Y)$ and verifies that $t_B = \mathsf{MAC}_{\bar{k}'}(B, X, Y)$. If the verification is successful, the client is convinced that it is communicating with the correct server: nobody other than the client and the relay could compute $xB = bX$. Both sides can use $k = \bar{k}$ for transmitting encrypted messages.

Tor currently uses SSL links to superencrypt its traffic. However, one can easily imagine ntor being steganographically encoded inside a covert channel, with the goal of circumventing censorship, on top of the original goals of anonymity, one-way authentication, and forward secrecy. This raises the question of how the curve points can be hidden. Using a pair of points xP, xP' in place of X does not work: it is even easier to distinguish from uniform than a single point. Möller's curve-or-twist approach, using a pair of points in place of the server's long-term key B, also does not work: the client is free to pick X on the curve or the twist, but the server

is then forced to pick Y on the same curve as X, and this is something visible to a censor.

Our solution applies straightforwardly to this protocol. It does not need twists; it ensures that the encodings of points on a single curve are uniformly distributed. More generally, our solution can be used to encode as many points as desired during one session or across sessions.

Our solution is also useful for several earlier applications: password-authenticated key exchange [59], ID-based encryption [56], pseudorandom-number generation [185, 186], and kleptography [328]. Note that [328] includes a treatment of covert channels and steganography, predating both StegoTorus and Telex.

6.3.4 Mapping strings to elliptic-curve points

Shallue and van de Woestijne at ANTS 2006 [288] gave a positive answer to the following question in pure algorithmic number theory: there is, provably, a (very easy) *probabilistic* polynomial-time algorithm that, given any elliptic curve E over any sufficiently large finite field $\mathbf{F}_q$, constructs a nonzero element of the group $E(\mathbf{F}_q)$; is there, provably, a *deterministic* polynomial-time algorithm for the same task? Shallue and van de Woestijne did more than construct a single point: they built a function $\phi : \mathbf{F}_q \to E(\mathbf{F}_q)$ such that each element of $E(\mathbf{F}_q)$ has at most C preimages, for a particular constant C; and they built a deterministic polynomial-time algorithm that computes $\phi(t)$ given q, E, and t. Trying $C + 1$ elements of $\mathbf{F}_q$ produces a nonzero element of $E(\mathbf{F}_q)$.

Some cryptographic protocols rely on hashing strings to curve points. For example, in the Boneh–Franklin identity-based encryption scheme [56], public keys are points on pairing-friendly curves, and a user with identity i has public key $H(i)$. Boneh and Franklin constructed a suitable function H for certain supersingular curves. Icart at Crypto 2009 [176] pointed out that the Shallue–van de Woestijne function ϕ produced suitable functions H for any elliptic curve, allowing the Boneh–Franklin system to be adapted to much more efficient non-supersingular (but still pairing-friendly) curves. Icart and subsequent authors explored various replacements ϕ for the Shallue–van de Woestijne function; see [176], [65], [136], [135], [128], [137], [133], and [129].

The reader should be wondering why one cannot simply define $\phi(t)$ by trying consecutive field elements x starting from t (for example, trying $x = t$, $x = t + 1$, $x = t + 2$, etc. in the prime-field case) until finding a curve point (x, y). The answer for Shallue and van de Woestijne is that this is not *proven* to take polynomial

time. On the other hand, there is overwhelming evidence for the conjecture that this takes polynomial time, and in cryptography there is no harm in making such a conjecture; cryptography is built on a foundation of conjectures that are much easier to question, such as the conjecture that there is no fast ECDL algorithm. This simple function from $\mathbf{F}_q$ to $E(\mathbf{F}_q)$ is exactly what is used in many papers on identity-based cryptography.

The real objection to this algorithm is that it does not take *constant* time. In many applications the time leaks secret information. One can compute the same function for almost all inputs in constant time by choosing an upper bound C on the number of consecutive field elements required with very high probability, say $C = 100$, and then always testing exactly C values of x; but testing C times for squares turns out to be a performance bottleneck.

It should be obvious that any of the functions ϕ mentioned above allows a public key $\phi(t) \in E(\mathbf{F}_q)$ to be represented as an element $t \in \mathbf{F}_q$, using a computation of $Q \mapsto \{t : \phi(t) = Q\}$ to encode a point and a computation of $t \mapsto \phi(t)$ to decode a point. However, this approach raises several important performance issues for key generation:

- The usual key-generation method produces a uniform random curve point by scalar multiplication, but there is no reason to think that this curve point can be expressed in the form $\phi(t)$. If it cannot then the user must try again, generating a new key.
- Even worse, to generate a *uniform* distribution of elements $t \in \mathbf{F}_q$ the user must generate a uniform random curve point Q, compute the number $k = \#\{t : \phi(t) = Q\}$ of preimages of the point, restart with probability $1 - k/C$, and finally select a uniform random preimage of the point. (This is what statisticians call "rejection sampling"; more advanced sampling methods are inapplicable since there is no way to generate a uniform random curve point with a specified value of k.) The average number of points generated is then $C\#E(\mathbf{F}_q)/q \approx C$. Applications that need real-time guarantees must budget for even more than C.
- Each of these C points takes time for not just a computation of Q but also a computation of $\{t : \phi(t) = Q\}$. This is the main bottleneck if ϕ is slow to invert.

In many protocols, ephemeral ECC keys are generated for every protocol run, and these computations can easily dominate the overall protocol performance. Of

course, even in protocols where key generation is rare, each use of t incurs the cost of computing $\phi(t)$.

6.3.5 Our contributions

We propose two ways to minimize these performance problems: Elligator 1 and Elligator 2. Elligator 1 takes an encoding function ϕ that is implicit in the paper [133] by Fouque, Joux, and Tibouchi. This function, in the cases we consider, maps $\{0, 1, 2, 3, \ldots, (q-1)/2\}$ *injectively* to $E(\mathbf{F}_q)$; see Theorem 11. Since the number of points on E over $\mathbf{F}_q$ is about q we have to try only about 2 points on average before finding a point of the form $\phi(t)$, and we do not need to randomize the preimage t. Not every elliptic curve is suitable for this function, but the curve requirements for Elligator 1 are compatible with state-of-the-art criteria for curve security and curve performance.

Fouque, Joux, and Tibouchi suggested that injectivity would be useful for a naive form of ElGamal encryption in which a message m is encoded injectively as an elliptic-curve point and then simply added to abP. We highlight the new anti-censorship application, namely encoding *points* as *strings* rather than encoding strings as points; we drastically simplify the definition of this function, while accelerating its forward and inverse computation; and we introduce a high-security high-speed elliptic curve that supports the function. See Section 6.6 for the curve, Section 6.5 for the function, and Section 6.4 for the cryptographic applications.

Elligator 2 introduces a new injective map ψ to elliptic curves. Elligator 2 has similar performance characteristics to Elligator 1. The main advantage of Elligator 2 is that it applies to more curves: in fact, every odd-characteristic elliptic curve that has a point of order 2, except for curves of j-invariant 1728. See Section 6.7 for this new map.

6.4 Elliptic-curve protocols

This section presents various elliptic-curve protocols in which public keys, ciphertexts, etc. are indistinguishable from uniform random strings. These protocols include all of the fundamental ECC constructions: static and ephemeral key exchange, encryption, and signatures.

The prerequisite for all of these protocols is an injective map ι from a set of strings $S \subseteq \{0,1\}^b$ to an elliptic-curve group $E(\mathbf{F}_q)$. We require $\#S$ to be very close

to 2^b, so that a uniform random element of S is indistinguishable from a uniform random b-bit string; see Section 6.4.6. We have two different methods to construct ι and S, as mentioned above; see Sections 6.5 and 6.7.

The idea, as explained in Section 6.1, is for each string $\tau \in S$ to represent the curve point $\iota(\tau)$, i.e., for the point $\iota(\tau)$ to be encoded as the string τ. A uniform random element of $\iota(S)$, encoded in this way, is indistinguishable from a uniform random b-bit string. We do not require $\iota(S)$ to be all of $E(\mathbf{F}_q)$; our protocols compensate for this by repeatedly generating curve points until finding elements of $\iota(S)$. For our constructions $\#\iota(S)$ is about $(1/2)\#E(\mathbf{F}_q)$, requiring only about 2 repetitions on average, as mentioned in Section 6.1.

Our primary objective in this section is to illustrate how easily ι can be used to systematically hide elliptic-curve points in a wide range of protocols. We do not mean to suggest that all of these protocols are being used in contexts where they need to be hidden from censors; but there is no obvious dividing line between protocols that would be useful for those contexts and protocols that would not.

Consider, for example, static ECDH public keys. A static public key might be distributed openly, as part of a cryptographic software package, in which case it does not add any new risk of censorship; changing the encoding of the public key does nothing to hide the package from a censor. On the other hand, if a client already has all necessary cryptographic software, and a server broadcasts a series of static public keys by encoding those keys inside otherwise innocent web pages or other cover traffic, then indistinguishability from uniform is exactly the tool needed to defend those public keys against a censor with an accurate model of the cover traffic. A censored Tor client should be able to see frequent updates of public keys for Tor bridges, for example; see [13] for a detailed discussion of the speed at which various governments detect and suppress access to Tor bridges.

6.4.1 Notation and domain parameters

To simplify the protocol statements we assume that the elliptic-curve group $E(\mathbf{F}_q)$ is cyclic: specifically, that it is generated by a standard base point P of order n. In the case that n is not prime (for example, n is 4 times a prime in Section 6.6) we do *not* restrict points to a prime-order subgroup: any proper subgroup is trivially distinguishable from the full group. For non-cyclic groups the protocols would have to be modified to handle multiple generators.

Let H denote a hash function, and let $||$ denote concatenation of strings. Symmetric authenticated encryption of a message M using a secret key k is denoted as

$c = \mathrm{Enc}_k(m)$, decryption as $m = \mathrm{Dec}_k(c)$. A standard security requirement for symmetric encryption is that the ciphertexts are indistinguishable from uniform (see, e.g., [32]); one can, for example, safely use AES in counter mode. For authentication there is a split in the literature between authenticators aiming at the "PRF" property, which guarantees indistinguishability, and authenticators aiming merely at the "MAC" property, which guarantees unforgeability but does not guarantee indistinguishability; we require the PRF property. See [276] for a unified security notion for authenticated encryption that guarantees indistinguishability from uniform random bit strings.

6.4.2 Long-term Diffie–Hellman keys

Each user U sets up a public key:

1. U generates a random integer u.
2. U computes $P_U = uP$. If $P_U \notin \iota(S)$ then U repeats from step 1.
3. U publishes $\iota^{-1}(P_U) = \tau_U$ and keeps u secret.

The encoding τ_U of the public key is a string which can be broadcast; it is indistinguishable from a random string.

Without further communication users Alice and Bob can compute a shared secret from their public strings τ_A and τ_B. Alice can compute this key as follows and then use it to authenticate and encrypt a message to Bob:

1. Alice computes $\iota(\tau_B) = P_B$.
2. Alice computes $k = H(u_A P_B)$.

Upon receiving an encrypted message from Alice, Bob can likewise compute the same shared secret and then decrypt the message and verify the authenticator:

1. Bob computes $\iota(\tau_A) = P_A$.
2. Bob computes $k = H(u_B P_A)$.

6.4.3 ElGamal encryption

Assume u_B and $P_B = u_B P$ is a static key pair of Bob and that Bob has published $\tau_B = \iota^{-1}(P_B)$. Alice wants to send Bob a message M. To encrypt the message she performs the following steps:

1. She generates a random integer r.
2. She computes the point $R = rP$. If $R \notin \iota(S)$ she repeats from step 1.
3. She computes $\iota^{-1}(R) = \tau_R$.

4. She computes $P_B = \iota(\tau_B)$.
5. She computes the shared key value $k = H(rP_B)$.
6. She encrypts message m using key k: $c = \mathrm{Enc}_k(m)$.
7. She sends the tuple (τ_R, c) as an encryption of m.

To decrypt the received message, Bob:

1. computes $\iota(\tau_R) = R$,
2. computes the same shared key value: $k = H(u_B R)$, and
3. decrypts the message: $m = \mathrm{Dec}_k(c)$.

ElGamal encryption also appears in [133] as an application of an injective map between strings and curve points, but our application is completely different from the application in [133]. There is a critical difference in the underlying encryption methods: we use symmetric cryptography to encrypt the message m using a key derived from rP_B, whereas [133] adds rP_B to a curve point M that represents m. This is exactly where [133] uses an injective map, namely to encode the string m as a curve point M. Note, however, that this also (1) prevents [133] from encrypting long messages and (2) allows malleability, in violation of the basic security standards for public-key encryption. We instead use the standard "KEM/DEM" structure [98] to provide secure public-key encryption, and as a consequence do not need to encode strings as curve points. We use an injective map for a completely different reason: we encode curve points as strings, preventing those points from being recognized by censors.

6.4.4 Short-term Diffie–Hellman keys

In this protocol, Alice and Bob agree on a shared secret without using long-term keys. This protocol is important as a way to provide forward secrecy. Of course, this provides no authentication, but authentication can be added as a subsequent layer.

1. Alice and Bob generate short-term keys as follows:
   (a) Generate a random integer r.
   (b) Compute point $R = rP$. If $R \notin \iota(S)$ repeat from step 1a.
2. Alice sends $\mu_A = \iota^{-1}(R_A)$ to Bob; Bob sends $\mu_B = \iota^{-1}(R_B)$ to Alice.
3. Alice decodes $R_B = \iota(\mu_B)$; Bob decodes $R_A = \iota(\mu_A)$.
4. Alice computes the shared key value $k = H(r_A R_B)$; Bob computes the same k as $H(r_B R_A)$.

6.4.5 Schnorr signatures

Assume that u_B and $P_B = u_B P$ are respectively private and public keys of Bob, where $P_B \in \iota(S)$ and Bob has published $\tau_B = \iota^{-1}(P_B)$. To sign a message m, Bob performs the following steps:

1. Choose a random integer r.
2. Compute $R = rP$. If $R \notin \iota(S)$, repeat from step 1.
3. Compute $\tau = \iota^{-1}(R)$.
4. Compute $h = H(\tau||\tau_B||m)$.
5. Compute $s = r + hu_B \pmod n$.
6. The signature is the tuple $\psi = (\tau, s)$.

The integer s must be encoded so as to be indistinguishable from uniform; see Section 6.4.6.

To verify the signature (τ, s), Alice performs the following steps:

1. Compute $P_B = \iota(\tau_B)$.
2. Compute $R = \iota(\tau)$.
3. Compute $h = H(\tau||\tau_B||m)$.
4. Compare $R + hP_B$ and sP. If they are equal, accept. Otherwise, reject.

Schnorr's original signature system actually sent (h, s). We follow "EdDSA" from [47] in sending an encoding of (R, s); security is the same, since one can reconstruct (h, s) from (R, s) and vice versa. The advantage of (R, s) is that it allows batching, making signature verification about twice as fast, as explained in [47]. Our use of ι makes this signature indistinguishable from a uniform random string.

We also follow EdDSA in including the public key τ_B in the hash. Robert Ransom has pointed out that if the public key is omitted from the hash then an attacker who does not know it can calculate it by first computing h and then computing $(sP - R)/h$, and as a consequence can see that two signatures come from the same public key. Ransom has also suggested a different way of hiding signatures from someone who does not know τ_B: namely, encrypting the signatures with a secret-key cipher keyed by a hash of τ_B. Of course, indistinguishability is unachievable against an attacker who *does* know τ_B: the attacker can simply verify the signature.

The second argument for including the public key τ_B in the hash is that it provides security against attacks focused on breaking several public keys instead of one. Recent paper by Bernstein [44] points out an error in the proof in [142] showing no decrease in security in classical Schnorr scheme if attacker focuses on keys of several users. Instead, a new proof is presented in [44] demonstrating that the

security of Schnorr scheme in multi-key scenario with a public key inserted into the hash (*key-prefixed variant* of the Schnorr's signature) is close to the security of the classical Schnorr scheme with only one targeted public key.

We encode R as τ both for transmission and for input to H. We could safely switch to a different encoding for input to H. In particular, we could use the same encoding used in EdDSA. This would make our signatures compatible with EdDSA signatures: anyone could convert an EdDSA signature into a signature of our form and vice versa.

6.4.6 Detecting differences from uniform

These protocols transmit uniform random elements of S. Usually S is not the set of *all* b-bit strings, so uniform random elements of S are not exactly uniform random b-bit strings. This is not a security problem if $\#S$ is very close to 2^b; we now analyze quantitatively what "very close" means.

Consider a channel that normally sends k independent uniform random b-bit strings. A user modifies the channel to instead transmit k independent uniform random elements of S. The censor's goal is to take action against the modified channel without taking action against the original channel.

The obvious strategy for the censor is to take action if and only if all of the b-bit strings are elements of S. This has no false negatives, but it produces a false positive whenever k independent uniform random b-bit strings happen to all be elements of S. If $\#S = 2^b - \delta$ then a false positive occurs with probability $(\#S/2^b)^k = (1 - \delta/2^b)^k \approx \exp(-k\delta/2^b)$. This strategy maximizes the difference in action probability between the original and modified channels; hence S^k is indistinguishable from $(\{0,1\}^b)^k$ when $k\delta/2^b$ is small.

Consider, for example, the NIST P-256 set $S = \left\{0, 1, 2, \ldots, 2^{256} - 2^{224} + 2^{192} + 2^{96} - 2\right\}$, with $b = 256$. Here $\delta \approx 2^{224}$. If the censor sees $k = 2^{35}$ sessions then $k\delta/2^b \approx 8$ so the censor will take action for the original channel with probability only about $\exp(-8) \approx 0.000335$. This may be an acceptable level of collateral damage.

As a second example illustrating the importance of the ratio $k\delta/2^b$, consider a set S of size $2^{256} - \delta$ where $\delta \leq 2^{160}$. This time a censor seeing $k \leq 2^{64}$ sessions has $k\delta/2^b \leq 2^{-32}$ so the probability is larger than 0.99999999; i.e., the censor will take action against practically every channel.

The encodings that we consider for elements of $E(\mathbf{F}_q)$ typically have $\#S = (q + 1)/2$. In each of our recommended examples, $\#S$ is between $2^b - 2^{b/2}$ and 2^b,

so $k\delta/2^b \leq k/2^{b/2}$. The probability difference is therefore negligible until k, the number of strings transmitted, grows to a noticeable fraction of $2^{b/2}$. This cannot be a concern when $E(\mathbf{F}_q)$ is chosen to resist standard discrete-logarithm attacks: those attacks take time only about $2^{b/2}$.

We also encode integers modulo $n = \#E(\mathbf{F}_q)$ as strings in Section 6.4.5. By Hasse's theorem, n is within $2\sqrt{q}$ of $q+1$, so if q is very close to 2^{b+1} then n is also forced to be very close to 2^{b+1}. Note that taking q slightly below 2^{b+1} does not force n to be below 2^{b+1}; for $n > 2^{b+1}$ we still use a $(b+1)$-bit encoding, restarting the protocol in Section 6.4.5 in the extremely unlikely case that an integer is 2^{b+1} or larger.

6.4.7 Active attacks

Censors can try replacing random-looking strings by other strings (possibly strings outside S) to see whether this has any visible effect. Presumably this replacement will *disrupt* communication, but protocol designers and implementors must take care to ensure that this replacement does not allow the censor to *detect* communication.

We briefly point out an attack against the password-authenticated key-exchange protocol of [59]; this attack is not in the censorship context, but it illustrates both the value of avoiding twists and the difficulty of protecting against active attacks. In the protocol of [59], Alice sends an encryption (using a shared password as a secret key) of either a point on a curve or a point on the twist of the curve; indistinguishability is important here for the ciphertext to avoid leaking information. Bob sends back two points, one on the curve and one on the twist; information is carried by the point on the same curve used by Alice, while the other point is random. Our attack is to actively rerandomize one of the two points sent by Bob. If this point is on the same curve then Alice aborts; if this point is not on the same curve then Alice does not notice and communication continues. This information leak allows the attacker to exclude half of all possible passwords, and repeating the attack allows the attacker to quickly find the right password.

6.5 Elligator 1: The injective map

Section 6.4 needs an injective map ι from a large set S of strings to $E(\mathbf{F}_q)$. Elligator 1 is one choice of ι; Elligator 2, introduced in Section 6.7, is another choice of ι. This

section presents the mathematical details of Elligator 1: the construction of ι, how to compute ι, how to test whether a curve point is in the image of ι, and how to invert ι on curve points in the image. To help the reader visualize the mathematical structure of ι we include a picture as Figure 6.1.

We impose certain requirements on q and E for Elligator 1: we consider only primes q; we require q to be congruent to 3 modulo 4; we require E to be a complete Edwards curve; and we impose an extra algebraic requirement ($c = 2/s^2$ in Theorem 9) that allows only half of all complete Edwards curves. Approximately $1/16$ of all isomorphism classes of elliptic curves over all finite fields satisfy these requirements. (Asymptotically 100% of all finite fields, ordered by size, are prime fields; 50% of those primes are congruent to 3 modulo 4; 25% of elliptic curves over those fields are complete Edwards curves; 50% of those satisfy the extra algebraic requirement.) See Section 6.6 for specific choices of q and E.

The heart of ι is a function $\phi : \mathbf{F}_q \to E(\mathbf{F}_q)$ defined in Theorem 9 and Definition 9. This function satisfies $\phi(t) = \phi(-t)$ for each $t \in \mathbf{F}_q$ but has no other collisions (see Theorem 10), so its restriction to $\{0, 1, 2, \ldots, (q-1)/2\}$ is injective. Theorem 11 simply defines S as $\{0, 1, 2, \ldots, (q-1)/2\}$ represented in little-endian form as b-bit strings, where $b = \lfloor \log_2 q \rfloor$, and defines ι as the corresponding representation of ϕ. For indistinguishability we add the requirement that $(q+1)/2$ be extremely close to 2^b.

As mentioned in Section 6.1, this function ϕ was introduced by Fouque, Joux, and Tibouchi in [133]. Our main contributions in this section are a much more concise definition of ϕ; much more direct proofs of the relevant properties of ϕ; a simple method to invert ϕ; and a simple test for whether a curve point is in the image of ϕ.

The function $-\phi$ has the same useful properties as ϕ. Our choice of sign is not the same as the choice in [133]: in the notation below, the ratio is $\chi(c)$, i.e., $\chi(2)$. This choice of sign slightly simplifies our formulas for the forward and inverse maps, although it is not the main simplification compared to [133]. We also comment that $\phi(t) = -\phi(1/t)$ for $t \neq 0$.

Computing ι in a sensible way is almost as fast as traditional point decompression, not a serious bottleneck compared to scalar multiplication. Inverting ι is slightly slower, but testing whether a curve point is in the image of ι is very fast. See Section 6.5.5 for further performance analysis.

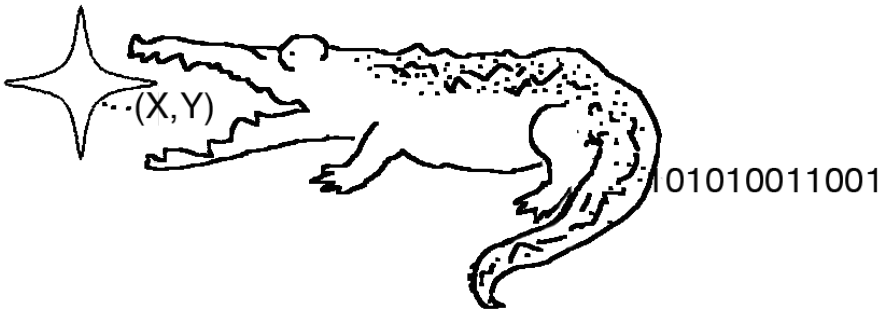

Figure 6.1: The structure of our encoding function ι^{-1}, a bijection from a large subset of $E(\mathbf{F}_q)$ to a large set S of b-bit strings. The elliptic curve E is required to be a complete Edwards curve, shown on the left together with a sample element $P = (x, y)$ of $E(\mathbf{F}_q)$. A sample b-bit output is shown on the right. See Theorems 9, 10, and 11 and Definition 9 for further details regarding the function in the middle.

6.5.1 Squares, square roots, and χ

Fix a prime power $q \equiv 3 \pmod 4$. In defining ι we consider only primes q for simplicity (so that $0, 1, 2, \ldots, (q-1)/2$ are distinct field elements) but our ϕ theorems also apply to prime powers.

Define $\chi : \mathbf{F}_q \to \mathbf{F}_q$ by $\chi(a) = a^{(q-1)/2}$. If a is a non-zero square then $\chi(a) = 1$; if a is a non-square then $\chi(a) = -1$; if $a = 0$ then $\chi(a) = 0$. Note that $(q-1)/2$ is odd since $q \equiv 3 \pmod 4$, so $\chi(-1) = -1$, so -1 is not a square. More generally, $\chi(\chi(a)) = \chi(a)$. There are several easy ways to manipulate χ arguments: for example, $\chi(ab) = \chi(a)\chi(b)$, $\chi(1/a) = \chi(a) = 1/\chi(a)$ if $a \neq 0$, and $\chi(a^2) = 1$ if $a \neq 0$.

If a is a square then $a^{(q+1)/4}$ is a square root of a: its square is $a^{(q+1)/2} = \chi(a)a = a$. More precisely, $a^{(q+1)/4}$ is the **principal** square root of a: the unique square root that is a square. Any square root b of a satisfies $b = \chi(b)a^{(q+1)/4}$.

The function χ is called a **quadratic character**. See [214] for further background on finite fields.

6.5.2 The map

Theorem 9. *Let q be a prime power congruent to 3 modulo 4. Let s be a nonzero element of $\mathbf{F}_q$ with $(s^2-2)(s^2+2) \neq 0$. Define $c = 2/s^2$. Then $c(c-1)(c+1) \neq 0$.*

Define $r = c + 1/c$ and $d = -(c+1)^2/(c-1)^2$. Then $r \neq 0$, and d is not a square. The following elements of $\mathbf{F}_q$ are defined for each $t \in \mathbf{F}_q \setminus \{\pm 1\}$:

$$\begin{aligned} u &= (1-t)/(1+t), \\ v &= u^5 + (r^2-2)u^3 + u, \\ X &= \chi(v)u, \\ Y &= (\chi(v)v)^{(q+1)/4}\chi(v)\chi\big(u^2 + 1/c^2\big), \\ x &= (c-1)sX(1+X)/Y, \\ y &= (rX - (1+X)^2)/(rX + (1+X)^2)\,. \end{aligned}$$

Furthermore $x^2 + y^2 = 1 + dx^2y^2$; $uvXYx(y+1) \neq 0$; and $Y^2 = X^5 + (r^2-2)X^3 + X$.

Proof. **$c(c-1)(c+1) \neq 0$:** By definition $c = 2/s^2$ so $c \neq 0$. By hypothesis $s^2 \neq 2$ and $s^2 \neq -2$ so $c \neq 1$ and $c \neq -1$.

$r \neq 0$: If $r = 0$ then $c = -1/c$ so $c^2 = -1$, contradiction.

d is not a square: Otherwise $-1 = d(c-1)^2/(c+1)^2$ is a square, contradiction.

u is defined and $u \neq 0$: By hypothesis $1 + t \neq 0$ and $1 - t \neq 0$.

$v \neq 0$: $r^2 - 2 = c^2 + 1/c^2$ so $v = u(u^2 + c^2)(u^2 + 1/c^2)$. If $v = 0$ then there are three possibilities: $u = 0$, contradiction; or $u^2 + c^2 = 0$ so $-1 = (u/c)^2$, contradiction; or $u^2 + 1/c^2 = 0$ so $-1 = (uc)^2$, contradiction.

$XY \neq 0$, so x is defined: As above $u^2 + 1/c^2 \neq 0$ so all factors in X and Y are nonzero.

$1 + X \neq 0$, so $x \neq 0$: If $X = -1$ then $u = -\chi(v)$ so $v = -\chi(v)(1 + r^2 - 2 + 1) = -\chi(v)r^2$ so $\chi(v) = -\chi(v)$, contradiction.

(X, Y) satisfies $Y^2 = X^5 + (r^2-2)X^3 + X$: $X = \chi(v)u$ so $X^5 + (r^2-2)X^3 + X = \chi(v)(u^5 + (r^2-2)u^3 + u) = \chi(v)v$. Also $\chi(v)v$ is a square so $(\chi(v)v)^{(q+1)/2} = \chi(v)v$ so $Y^2 = \chi(v)v$.

$rX + (1+X)^2 \neq 0$, so y is defined: If $rX = -(1+X)^2$ then $(r^2 + 4r)X^2 = X^4 - 2X^2 + 1$ so

$$\begin{aligned} Y^2 &= X(X^4 + (r^2-2)X^2 + 1) = X^3(2r^2 + 4r) \\ &= rX \cdot X^2(2r + 4) = -(1+X)^2X^2(s + 2/s)^2 \end{aligned}$$

so -1 is a square, contradiction.

$y + 1 \neq 0$: If $y = -1$ then $(rX - (1+X)^2)/(rX + (1+X)^2) = -1$ so $rX - (1+X)^2 = -(rX + (1+X)^2)$ so $rX = 0$, contradiction.

$\boldsymbol{x^2 + y^2 = 1 + dx^2y^2}$**:** First $(c-1)^2s^2 = (c-1)^2(2/c) = 2(r-2)$ so

$$\begin{aligned} Y^2(1-x^2) &= Y^2 - (c-1)^2s^2X^2(1+X)^2 \\ &= X^5 + (r^2-2)X^3 + X - 2(r-2)X^2(1+X)^2 \\ &= X(rX - (1+X)^2)^2. \end{aligned}$$

Similarly $-d = (c+2+1/c)/(c-2+1/c) = (r+2)/(r-2)$ so $-d(c-1)^2s^2 = 2(r+2)$ so

$$\begin{aligned} Y^2(1-dx^2) &= Y^2 - d(c-1)^2s^2X^2(1+X)^2 \\ &= X^5 + (r^2-2)X^3 + X + 2(r+2)X^2(1+X)^2 \\ &= X(rX + (1+X)^2)^2. \end{aligned}$$

Note that $Y^2(1-dx^2) \neq 0$, and divide: $(1-x^2)/(1-dx^2) = (rX-(1+X)^2)^2/(rX+(1+X)^2)^2 = y^2$; i.e., $x^2+y^2 = 1+dx^2y^2$. □

Definition 9. *In the situation of Theorem 9, the* **decoding function** *for the complete Edwards curve $E : x^2 + y^2 = 1 + dx^2y^2$ is the function $\phi : \mathbf{F}_q \to E(\mathbf{F}_q)$ defined as follows: $\phi(\pm 1) = (0,1)$; if $t \notin \{\pm 1\}$ then $\phi(t) = (x,y)$.*

6.5.3 Inverting the map

Theorem 10. *In the situation of Definition 9:*

1. *If $t \in \mathbf{F}_q$ then the set of preimages of $\phi(t)$ under ϕ is $\{t, -t\}$.*
2. *$\phi(\mathbf{F}_q)$ is the set of $(x,y) \in E(\mathbf{F}_q)$ such that*
   - *$y + 1 \neq 0$;*
   - *$(1+\eta r)^2 - 1$ is a square, where $\eta = \dfrac{y-1}{2(y+1)}$; and*
   - *if $\eta r = -2$ then $x = 2s(c-1)\chi(c)/r$.*
3. *If $(x,y) \in \phi(\mathbf{F}_q)$ then the following elements $\bar{X}, z, \bar{u}, \bar{t}$ of $\mathbf{F}_q$ are defined and $\phi(\bar{t}) = (x,y)$:*

$$\begin{aligned} \bar{X} &= -(1+\eta r) + ((1+\eta r)^2 - 1)^{(q+1)/4}, \\ z &= \chi\big((c-1)s\bar{X}(1+\bar{X})x(\bar{X}^2 + 1/c^2)\big), \\ \bar{u} &= z\bar{X}, \\ \bar{t} &= (1-\bar{u})/(1+\bar{u})\,. \end{aligned}$$

Proof. Statement 1 of the theorem has two parts: a forward statement $\phi(t) = \phi(-t)$, and a reverse statement that there are no other preimages. Statement 2

also has two parts: a forward statement that any $(x,y) \in \phi(\mathbf{F}_q)$ satisfies certain conditions, and a reverse statement that any element of $E(\mathbf{F}_q)$ satisfying those conditions is in $\phi(\mathbf{F}_q)$. We organize the proof as (A) forward 1; (B) 3, forward 2, and reverse 1; (C) reverse 2.

A. Fix $t \in \mathbf{F}_q$. We now show that $\phi(t) = \phi(-t)$. This is the forward part of statement 1 in the theorem.

If $t \in \{\pm 1\}$ then $\phi(t) = (0,1) = \phi(-t)$ by definition. Assume from now on that $t \notin \{\pm 1\}$.

Define u, v, X, Y, x, y from t as in Theorem 9. Then $\phi(t) = (x,y)$ by definition.

Put $t' = -t$, and define u', v', X', Y', x', y' the same way from t'. Then $\phi(t') = (x', y')$. The proof strategy is to compare successively u' to u, v' to v, etc., concluding that $x' = x$ and $y' = y$.

$\boldsymbol{u'} = (1-t')/(1+t') = (1+t)/(1-t) = 1/u$.

$\boldsymbol{v'} = u'^5 + (r^2-2)u'^3 + u' = \frac{1}{u^5} + (r^2-2)\frac{1}{u^3} + \frac{1}{u}$, so $v'u^6 = u + (r^2-2)u^3 + u^5 = v$; i.e., $v' = v/u^6$. Note that $\chi(v') = \chi(v)$ since $\chi(u^6) = 1$.

$\boldsymbol{X'} = \chi(v')u' = \chi(v)/u = 1/(\chi(v)u) = 1/X$ since $\chi(v) = 1/\chi(v)$.

$\boldsymbol{y'} = \dfrac{rX'-(1+X')^2}{rX'+(1+X')^2} = \dfrac{r\frac{1}{X}-(1+\frac{1}{X})^2}{r\frac{1}{X}+(1+\frac{1}{X})^2} = \dfrac{rX-(X+1)^2}{rX+(X+1)^2} = y$.

$\boldsymbol{Y'} = (\chi(v')v')^{(q+1)/4}\chi(v')\chi\big(u'^2 + 1/c^2\big)$. This takes the most work; the first and third factors each need careful analyses. The second factor is easy: $\chi(v') = \chi(v)$ as above.

First factor: $\chi(v')v' = \chi(v)v/u^6$. Note that the product $\chi(u)u^3$ is a square and is therefore the principal square root of u^6; i.e., $(u^6)^{(q+1)/4} = \chi(u)u^3$. Hence $(\chi(v')v')^{(q+1)/4} = (\chi(v)v/u^6)^{(q+1)/4} = (\chi(v)v)^{(q+1)/4}\chi(u)/u^3$.

Third factor: Recall that $v = u(u^2+c^2)(u^2+1/c^2)$ and that $\chi(a) = \chi(ab^2)$ for any $b \neq 0$. Thus

$$\begin{aligned}\chi\big(u'^2 + 1/c^2\big) &= \chi\big(c^2u^4(u'^2 + 1/c^2)(u^2+1/c^2)^2\big)\\ &= \chi\big(u^2(c^2+u^2)(u^2+1/c^2)^2\big)\\ &= \chi\big(uv(u^2+1/c^2)\big).\end{aligned}$$

Now multiply to obtain $Y' = Y\chi(u)\chi(uv)/u^3 = Y/(\chi(v)u)^3 = Y/X^3$. Finally

$$\begin{aligned}\boldsymbol{x'} &= (c-1)sX'(1+X')/Y' = (c-1)s\frac{1}{X}\left(1+\frac{1}{X}\right)\Big/\frac{Y}{X^3}\\ &= (c-1)sX(1+X)/Y = x.\end{aligned}$$

Hence $\phi(-t) = (x', y') = (x, y) = \phi(t)$ as claimed.

B. Fix $t \in \mathbf{F}_q$, and define $(x,y) = \phi(t)$. We show that $\bar{X}, z, \bar{u}, \bar{t}$ in the theorem are defined and that $\bar{t} \in \{t, -t\}$, so $\phi(\bar{t}) = (x,y)$; this is statement 3 in the theorem.

We also show the forward part of statement 2: namely, $y+1 \neq 0$; $(1+\eta r)^2 - 1$ is a square, where $\eta = (y-1)/(2(y+1))$; and if $\eta r = -2$ then $x = 2s(c-1)\chi(c)/r$. We also show the reverse part of statement 1: there are no preimages of $\phi(t)$ other than t and $-t$.

The definition of ϕ has two cases: if $t \in \{1,-1\}$ then $(x,y) = (0,1)$; if $t \notin \{1,-1\}$ then u, v, X, Y, x, y are defined in Theorem 9. Note that in the second case $x \neq 0$ by Theorem 9, so in particular t is not a preimage of $(0,1)$.

In the first case $y+1 = 2 \neq 0$; $\eta = 0$; $(1+\eta r)^2 - 1 = 0$ is a square; $\bar{X} = -1$; $z = 0$; $\bar{u} = 0$; and $\bar{t} = 1 \in \{t, -t\}$. As noted above, 1 and -1 are the only preimages of $(0,1)$.

What remains is the second case. Here $y+1 \neq 0$ by Theorem 9. The equation $y = (rX-(1+X)^2)/(rX+(1+X)^2)$ implies $X^2+(2+r(y-1)/(y+1))X+1 = 0$, i.e., $X^2+2(1+\eta r)X+1 = 0$. Note that this forces the discriminant $4(1+\eta r)^2-4$ to be a square; i.e., $(1+\eta r)^2-1$ is a square. Divide by X to see that $X+1/X = -2(1+\eta r)$.

If $\eta r = -2$ then $(X-1)^2 = 0$ so $X = 1$ so $u \in \{\pm 1\}$; the case $u = -1$ forces $1-t = -(1+t)$, contradiction, so $u = 1$ and $t = 0$; so $v = r^2$, so $Y = (r^2)^{(q+1)/4}\chi(1+1/c^2) = \chi(r)r\chi(r/c) = r\chi(c)$, so $x = 2(c-1)s\chi(c)/r$ as claimed; also note for future reference that $y = (r-4)/(r+4)$, i.e., $\phi(0) = (2(c-1)s\chi(c)/r, (r-4)/(r+4))$.

Define $t' = -t$, and define u', v', X', Y', x', y' as in Part A of this proof. Then $X' = 1/X$, so $X + X' = -2(1+\eta r)$.

By construction $1+\eta r+\bar{X}$ is a square root of $(1+\eta r)^2-1$; i.e., $\bar{X}^2+2(1+\eta r)\bar{X}+1 = 0$. Now $(\bar{X}-X)(\bar{X}-X') = \bar{X}^2-(X+X')\bar{X}+XX' = \bar{X}^2+2(1+\eta r)\bar{X}+1 = 0$ so $\bar{X} = X$ or $\bar{X} = X'$. This forces $\bar{u} = u$ or $\bar{u} = u'$, since the definition of z turns out to match $\chi(v)$ and $\chi(v')$:

- If $\bar{X} = X$ then $(c-1)s\bar{X}(1+\bar{X}) = xY$ so $z = \chi\big(x^2Y(X^2+1/c^2)\big) = \chi(Y)\chi\big(X^2+1/c^2\big)$. Note that $(\chi(v)v)^{(q+1)/4}$ is a square and $\chi\big(u^2+1/c^2\big) = \chi\big(X^2+1/c^2\big)$, so $\chi(Y) = \chi(v)\chi(X^2+1/c^2)$, so $z = \chi(v)$, so $\bar{u} = \chi(v)X = u$, so $\bar{t} = t$.
- If $\bar{X} = X'$ then similarly $z = \chi(v')$, $\bar{u} = u'$, and $\bar{t} = t' = -t$.

To summarize, $\bar{t} \in \{t, -t\}$, so $\phi(\bar{t}) = (x,y)$.

The same logic also shows that there are no preimages p of (x,y) except for t and $-t$. Indeed, if $(x,y) = \phi(p)$ then substituting p for t in the same proof shows that $\bar{t} \in \{p, -p\}$, so $p \in \{\bar{t}, -\bar{t}\} = \{t, -t\}$.

C. Fix $(x,y) \in E(\mathbf{F}_q)$. Assume that $y+1 \neq 0$; that $(1+\eta r)^2 - 1$ is a square, where $\eta = (y-1)/(2(y+1))$; and that if $\eta r = -2$ then $x = 2s(c-1)\chi(c)/r$. We now show that $(x,y) \in \phi(\mathbf{F}_q)$. This is the reverse part of statement 2 of the theorem.

If $x = 0$ then $(x,y) = (0,\pm 1)$ from the curve equation; but $y+1 \neq 0$, so $(x,y) = (0,1) = \phi(1) \in \phi(\mathbf{F}_q)$ as claimed. Assume from now on that $x \neq 0$.

If $y = 1$ then $x = 0$ from the curve equation, contradiction. Hence $y \neq 1$; i.e., $\eta \neq 0$.

Define $X = -(1+\eta r) + ((1+\eta r)^2 - 1)^{(q+1)/4}$. As above $1 + \eta r + X$ is a square root of $(1+\eta r)^2 - 1$, so $X^2 + 2(1+\eta r)X + 1 = 0$. This quadratic equation has several consequences. First, $X \neq 0$. Second, $rX + (1+X)^2 \neq 0$: otherwise subtract to see that $(1-2\eta)rX = 0$, so $1 = 2\eta$, so $y - 1 = y + 1$, contradiction. Third, $X \neq -1$: otherwise $\eta = 0$, contradiction. Fourth, $y = (rX - (1+X)^2)/(rX + (1+X)^2)$.

If $X = 1$ then $y = (r-4)/(r+4)$; also $\eta r = -2$ so by assumption $x = 2s(c-1)\chi(c)/r$ so $(x,y) = \phi(0) \in \phi(\mathbf{F}_q)$. Assume from now on that $X \neq 1$.

Observe that

$$\begin{aligned}
&(rX + (1+X)^2)^2(1-y^2) \\
&\quad = (rX + (1+X)^2)^2 - (rX - (1+X)^2)^2 \\
&\quad = 4rX(1+X)^2.
\end{aligned}$$

Recall that $-d = (r+2)/(r-2)$ and similarly observe that

$$\begin{aligned}
&(rX + (1+X)^2)^2(1-dy^2) \\
&\quad = (rX + (1+X)^2)^2 + \frac{r+2}{r-2}(rX - (1+X)^2)^2 \\
&\quad = (2r/(r-2))(X^4 + (r^2-2)X^2 + 1).
\end{aligned}$$

Note that $1 - dy^2 \neq 0$ since d is not a square. Divide:

$$x^2 = \frac{1-y^2}{1-dy^2} = \frac{2(r-2)X^2(1+X)^2}{X^5 + (r^2-2)X^3 + X}.$$

Define $Y = (c-1)sX(1+X)/x$. Then

$$\begin{aligned}
Y^2 &= (c-1)^2 s^2 X^2 (1+X)^2 / x^2 \\
&= 2(r-2)X^2(1+X)^2/x^2 \\
&= X^5 + (r^2-2)X^3 + X.
\end{aligned}$$

Define $z = \chi\big(Y(X^2 + 1/c^2)\big)$. Both Y and $X^2 + 1/c^2$ are nonzero, so $z \in \{\pm 1\}$. Define $u = zX$. Then $u \in \{\pm X\}$. Note that $u \neq -1$, since $X \notin \{\pm 1\}$.

Define $v = u^5 + (r^2 - 2)u^3 + u$. Then $v = z(X^5 + (r^2 - 2)X^3 + X) = zY^2$, so $\chi(v) = \chi(z) = z$. Hence $X = \chi(v)u$ and $Y^2 = \chi(v)v$. Furthermore $\chi(v) = z = \chi\big(Y(X^2 + 1/c^2)\big) = \chi\big(Y(u^2 + 1/c^2)\big)$, so $\chi(Y) = \chi(v)\chi\big(u^2 + 1/c^2\big)$, so $Y = (\chi(v)v)^{(q+1)/4}\chi(v)\chi\big(u^2 + 1/c^2\big)$.

Finally define $t = (1 - u)/(1 + u)$. Then $t \notin \{\pm 1\}$ and $u = (1 - t)/(1 + t)$. The formulas for u, v, X, Y, x, y in Theorem 9 are all satisfied, so $(x, y) = \phi(t) \in \phi(\mathbf{F}_q)$ as claimed. □

6.5.4 Encoding as strings

Theorem 11. *In the situation of Definition 9, assume that q is prime, and define $b = \lfloor \log_2 q \rfloor$. Define $\sigma : \{0,1\}^b \to \mathbf{F}_q$ by $\sigma(\tau_0, \tau_1, \ldots, \tau_{b-1}) = \sum_i \tau_i 2^i$. Define $S = \sigma^{-1}(\{0, 1, 2, \ldots, (q-1)/2\})$. Define $\iota : S \to E(\mathbf{F}_q)$ as follows: $\iota(\tau) = \phi(\sigma(\tau))$. Then $\#S = (q+1)/2$; ι is an injective map from S to $E(\mathbf{F}_q)$; and $\iota(S) = \phi(\mathbf{F}_q)$.*

Proof. First $2^b \leq q$ so the integers $0, 1, \ldots, 2^b - 1$ are distinct in $\mathbf{F}_q$; hence σ is injective. Furthermore $2^b > q/2$ so $\{0, 1, \ldots, (q-1)/2\}$ is a subset of $\{0, 1, \ldots, 2^b - 1\}$; hence each of $0, 1, \ldots, (q-1)/2$ has a preimage under σ, and S has exactly $(q+1)/2$ elements.

If $\iota(\tau) = \iota(\tau')$ then $\phi(\sigma(\tau)) = \phi(\sigma(\tau'))$, so $\sigma(\tau) = \pm\sigma(\tau')$ by Theorem 10; but $\sigma(\tau)$ and $\sigma(\tau')$ are both in $\{0, 1, \ldots, (q-1)/2\}$, so $\sigma(\tau) = \sigma(\tau')$, so $\tau = \tau'$. Hence ι is injective.

Each element of $\iota(S)$ has the form $\phi(\sigma(\tau))$ and is therefore in $\phi(\mathbf{F}_q)$. Conversely, if $P \in \phi(\mathbf{F}_q)$ then $P = \phi(t)$ for some $t \in \mathbf{F}_q$, so also $P = \phi(-t)$ by Theorem 10. At least one of $t, -t$ is in $\{0, 1, \ldots, (q-1)/2\}$, i.e., in $\sigma(S)$, so P is in $\phi(\sigma(S)) = \iota(S)$. □

6.5.5 Performance analysis

The definitions of u, v, X, Y, x, y in Theorem 9 involve divisions by $1 + t$, c, Y, and $rX + (1 + X)^2$. The reciprocal of c is trivially precomputed, and the other divisions are easily replaced by a few multiplications: one simply stores field elements as fractions, i.e., works in projective coordinates. There are several easy ways to reduce the number of multiplications: for example, factor $u^5 + (r^2 - 2)u^3 + u$ as $u(u^2 + c^2)(u^2 + 1/c^2)$, and reuse $u^2 + 1/c^2$ in computing Y.

The main bottlenecks are then the following exponentiations: computing $\chi(v)$ (used repeatedly); computing $\chi\big(u^2 + 1/c^2\big)$; computing $(\chi(v)v)^{(q+1)/4}$, the principal square root of $\chi(v)v$; and computing a final division if the output (x, y) is needed in

affine coordinates instead of projective coordinates. The only essential exponentiation is for the square-root computation: the χ computations and division can use Euclid's algorithm (blinded to protect against timing attacks) rather than exponentiation.

Similar comments apply to inverting ϕ (or ι). There is one essential exponentiation, the square-root computation to obtain $\bar{X}$ in Theorem 10. There are also two exponentiations that can be replaced by Euclidean computations: one χ computation to obtain z, and one division to obtain the final output $\bar{t}$. Fractions eliminate the initial division by $2(y+1)$, but fractions cannot be used for $\bar{t}$, since the goal is to obtain the unique string representing $\bar{t}$.

It is easier to test, given $(x,y) \in \mathbf{F}_q \times \mathbf{F}_q$, whether $(x,y) \in \phi(\mathbf{F}_q)$ (i.e., whether $(x,y) \in \iota(S)$) without inverting ϕ. One first checks $x^2 + y^2 = 1 + dx^2y^2$ to verify $(x,y) \in E(\mathbf{F}_q)$, if this is not already known. Then, by Theorem 10, $(x,y) \in \phi(\mathbf{F}_q)$ if and only if the following three conditions are satisfied:

- $y + 1 \neq 0$;
- $(1+\eta r)^2 - 1$ is a square, where $\eta = (y-1)/(2(y+1))$; i.e., $r(y-1)(r(y-1) + 4(y+1))$ is a square;
- if $\eta r = -2$ (equivalently, if $r(y-1) = -4(y+1)$) then $x = 2s(c-1)\chi(c)/r$.

This requires a few multiplications and one χ computation.

6.6 Construction of a suitable elliptic curve for Elligator 1

This section introduces a new high-security high-speed elliptic curve, called Curve1174, that supports the injective map presented in Section 6.5. In particular, this section specifies Curve1174; presents the criteria that we used to construct Curve1174; and analyzes the extent to which various previous curves meet the same criteria.

Curve1174 illustrates state-of-the-art criteria for choosing elliptic curves. It is compatible with Elligator 1, and is also compatible with the new map Elligator 2 introduced in the next section. It is even more concisely expressible than the existing Curve25519 [43] curve.

We do not mean to suggest that users of Curve25519 are required to switch to Curve1174 to take advantage of this paper. Curve25519 is fully compatible with Elligator 2.

6.6.1 The curve

Curve1174 is the Edwards curve $x^2 + y^2 = 1 - 1174x^2y^2$ over the field $\mathbf{F}_q$, where q is the prime number $2^{251} - 9$. The coefficient -1174 is a non-square in $\mathbf{F}_q$, so Curve1174 is a complete Edwards curve by [49, Theorem 3.3]: the sum of any two points (x_1, y_1) and (x_2, y_2) in Curve1174($\mathbf{F}_q$) is

$$\left(\frac{x_1y_2 + y_1x_2}{1 - 1174x_1x_2y_1y_2}, \frac{y_1y_2 - x_1x_2}{1 + 1174x_1x_2y_1y_2}\right),$$

with no divisions by 0 and no exceptional cases. The neutral element of the curve is $(0, 1)$.

To see that we have the desired injective map, note that q is congruent to 3 modulo 4; define s as the element

$$1806494121122717992522804053500797229 6$$
$$4843876698553887124072201084993488642 1$$

of $\mathbf{F}_q$ (split onto two lines here for readability); define $c = 2/s^2$; and define $d = -(c+1)^2/(c-1)^2$. Then $d = -1174$. The Edwards curve in Theorem 9 and Definition 9, for this choice of (q, s), is exactly Curve1174.

Curve1174 is birationally equivalent to the Montgomery curve $(4/1175)V^2 = U^3 + (4/1175 - 2)U^2 + U$ by [49, Theorem 3.2]. The leading coefficient $4/1175$ is a non-square in $\mathbf{F}_q$, so $V^2 = U^3 + (4/1175 - 2)U^2 + U$ is a nontrivial quadratic twist of this curve.

The number of points on the twist is thus $q + 1 + t$, and the number of points on Curve1174 is $q + 1 - t$, where $t = 45330879683285730139092453152713398836$. These integers $q + 1 + t, q + 1 - t$ have the form $4p_0, 4p_1$ respectively, where p_0 and p_1 are primes close to 2^{249}. Generic methods to compute a discrete logarithm on Curve1174 or its twist take approximately $\sqrt{\pi 2^{247}} \approx 2^{124.3}$ group operations on average.

The point $(U, V) = (4, 1922577764211167023040871244220551478340301270840905838377461328496334409 6)$ on the Montgomery curve $(4/1175)V^2 = U^3 + (4/1175 - 2)U^2 + U$ has order $4p_1$. The corresponding point on Curve1174 is $(x, y) = (4/V, 3/5)$.

Curve1174 and its twist do not have any structure allowing fast pairings or other special approaches to computing discrete logarithms. The primes p_0 and p_1 do not equal the field characteristic q. The order of q modulo p_0 is not small: it is $(p_0-1)/2$. The order of q modulo p_1 is not small: it is p_1-1. The endomorphism ring of Curve1174 has a large discriminant: up to squares this discriminant equals t^2-4q, which is divisible once by the prime 1615674151140249923338703492557 99, so the discriminant must be a multiple of this prime.

6.6.2 Design criteria

We consider only prime fields. Bernstein, citing subfield attacks from [140] and [110], wrote in [43] that prime fields "have the virtue of minimizing the number of security concerns for elliptic-curve cryptography"; see [130] and [255] for recent developments of the attack strategy from [110]. Similarly, the Brainpool standard [60] and NSA's Suite B standards [206] require prime fields. There is general agreement that prime fields are the safe, conservative choice for ECC. Prime fields also perform very well across a wide range of processors; the current ECC speed records on high-end Intel processors take advantage of special Intel support for binary fields (see [243]), but most CPUs do not have any comparable support.

We consider only primes q congruent to 3 modulo 4. This is required for the injective map ϕ. These primes also have the well-known benefit of allowing very simple square-root computations; most other primes allow square-root computations at similar *speed* but with more complicated methods.

We consider only complete Edwards curves, i.e., curves $x^2+y^2=1+dx^2y^2$ where d is not a square. This is required for the injective map. About 25% of all elliptic curves over $\mathbf{F}_q$ are expressible as complete Edwards curves, as mentioned in [49, Abstract] and experimentally verified in [45, Section 4]. Complete Edwards curves also have the advantages of being extremely fast and of allowing a single addition formula with no exceptions. Complete Edwards curves are also expressible as Montgomery curves supporting very fast and uniform Montgomery-ladder computations.

To protect against generic discrete-logarithm algorithms we impose the standard requirement of a large prime dividing the number of curve points. This forces q to be even larger, where the gap accounts for the cofactor: the number of curve points divided by this prime. To minimize the performance problems of a large q we consider only Edwards curves with minimal cofactor, namely 4. The number of

curve points is 4 times a prime for slightly below 1% of all choices of d, for the size of q that we consider below.

We also impose the requirement of "twist security": a large prime dividing the number of points on the quadratic twist of the curve. This prevents "twist attacks" against protocols that use the "Montgomery ladder" without checking that incoming points are on the curve; this defense was proposed by Bernstein in [42]. For $q \equiv 3 \pmod 4$ roughly $1/10000$ of all choices of d have the number of points on the curve and the number of points on the twist each being 4 times a prime.

We require d to have the form $-(c+1)^2/(c-1)^2$ with $c = 2/s^2$. This is required for the injective map, and covers about half of all non-squares d for $q \equiv 3 \pmod 4$.

For standard performance reasons we take q very close to, but not above, a power of 2. The primes $q \equiv 3 \pmod 4$ within 32 of 2^e for $200 \leq e \leq 300$ are $2^{206} - 5$, $2^{212} - 29$, $2^{226} - 5$, $2^{243} - 9$, $2^{251} - 9$, and $2^{285} - 9$. Note that these fields ensure that σ^{-1} covers nearly all of $\{0, 1, \ldots, 2^b - 1\}$ giving a very close to uniform distribution of the encoding function. We focus on the last two of these primes as providing quantitatively safe security levels, and choose $2^{251} - 9$ as being obviously faster.

Some curve operations involve multiplications by d. To speed up these multiplications we take the smallest possible d in absolute value, subject to the other requirements. The choice $d = -1174$ for $q = 2^{251} - 9$ is smaller than expected.

6.6.3 Previous curves over prime fields

There is a long history of specific elliptic curves being designed to meet various security and performance criteria. For example, almost fifteen years ago the IEEE P1363 standard [178, Sections A.9–A.12]

- specified curves $y^2 = x^3 - 3x + b$ to "provide the fastest arithmetic on elliptic curves";
- imposed various further conditions upon these curves, with the security goal of making discrete logarithms difficult to compute; and
- specified a procedure to generate "verifiably pseudo-random" curves meeting these conditions.

NIST's standard curves P-192, P-224, P-256, P-384, and P-521 were generated as follows: five particular prime fields were chosen with the goal of maximizing performance; the IEEE P1363 procedure was used to generate one curve over each of those fields.

Subsequent research developed new security and performance criteria for curves over prime fields: twist security, Montgomery compatibility, Edwards compatibility, and completeness. The NIST curves, unsurprisingly, flunk these criteria: choosing cofactor 1 for $y^2 = x^3 - 3x + b$ is incompatible with both Montgomery and Edwards, and one cannot expect twist security if it is not demanded in advance. Newer curves meet all of these criteria: for example, Curve25519 was explicitly designed for twist security and Montgomery compatibility, and was shown in [49] to also be expressible as a complete Edwards curve.

These extra criteria do not improve discrete-logarithm security, but they do improve real-world security. They allow the *simplest* implementations to be *correct* implementations, whereas for other curves the simplest implementations that seem to work actually have hidden flaws that compromise security. See, e.g., [53, Section 4.1], [181], and [134].

We are imposing a new security condition to support censorship circumvention: namely, an efficient way to encode a large fraction of all curve points as strings indistinguishable from uniform random strings. The generality of Elligator 1 makes it easy to imagine how this security condition could be accidentally met by previously generated curves:

- The advantages of complete Edwards curves have been well known for five years.
- The advantages of Montgomery curves have been well known for even longer. A random Montgomery curve has a good chance of being expressible as a complete Edwards curve; see [45, Section 4].
- All complete Edwards curves over $\mathbf{F}_q$ for $q \equiv 3 \pmod 4$ meet the new security condition. Half of these curves are within the streamlined case expressed by Theorem 9.

Given the amount of speed optimization of Curve25519 (see [43], [144], [97], [47], and [50]) and the wide deployment of Curve25519 in several applications (see, e.g., [14]) one of our initial goals for this paper was to show that Curve25519 meets this security condition. However, Elligator 1 is clearly limited to $q \equiv 3 \pmod 4$, while Curve25519 is defined over $\mathbf{F}_q$ with $q \equiv 1 \pmod 4$. We provide two different solutions for this problem: Elligator 2 (see Section 6.7) and Curve1174.

We would expect serious implementations of Curve1174 to be competitive in speed with Curve25519. Curve1174 has some small advantages: for example, $2^{251} - 9$ is closer to a power of 2 than $2^{255} - 19$ is; 1174 is considerably smaller than 486662;

and square roots modulo $2^{251}-9$ are slightly easier than square roots modulo $2^{255}-19$. On the other hand, Curve25519 also has a small advantage: it is expressible in "-1-twisted Edwards form", allowing the speedup explained in [171]. Obviously applications already using Curve25519 should not switch away from it, but for new applications it is not clear which curve is better.

6.7 Elligator 2: handling generic curves with a point of order 2

This section introduces a new injective map ψ to any elliptic curve of the form $y^2 = x^3 + Ax^2 + Bx$ with $AB(A^2 - 4B) \neq 0$ over any odd finite field, i.e., any finite field of odd characteristic. We emphasize that the characteristic is not required to be 3 modulo 4. This curve shape includes all Montgomery curves $y^2 = x^3 + Ax^2 + x$ except $y^2 = x^3 + x$, and in particular it includes Curve25519.

Any curve of this form has a point $(0, 0)$ of order 2. Conversely, over any odd finite field, almost every elliptic curve having a point of order 2 can be written in this form. Which means this map has better support of standardized curves.

Indeed, an elliptic curve over an odd finite field can always be written as $y^2 = u^3 + a_2u^2 + a_4u + a_6$; a point of order 2 on this curve must have the form $(r, 0)$ where $r^3 + a_2r^2 + a_4r + a_6 = 0$; substituting $u = x + r$ produces the curve $y^2 = x^3 + Ax^2 + Bx$ where $A = a_2 + 3r$ and $B = a_4 + 2a_2r + 3r^2$. This curve must have $B(A^2 - 4B) \neq 0$ since it is elliptic. The only exceptional case is $A = 0$, i.e., curves whose j-invariant equals 1728; this section assumes $A \neq 0$.

6.7.1 Squares

Fix an odd prime power q. As in Section 6.5.1 we define $\chi : \mathbf{F}_q \to \mathbf{F}_q$ by $\chi(a) = a^{(q-1)/2}$, and we have $\chi(a)$ equal to 1, -1, or 0 when a is, respectively, a non-zero square, a non-square, or zero.

The definition of Elligator 2 is parametrized by a **square-root function** for $\mathbf{F}_q$: a function $\sqrt{\ } : \mathbf{F}_q^2 \to \mathbf{F}_q$ such that $\sqrt{a^2} \in \{a, -a\}$ for each $a \in \mathbf{F}_q$, where $\mathbf{F}_q^2$ means $\left\{a^2 : a \in \mathbf{F}_q\right\}$. Note that a square-root function $\sqrt{\ }$ is completely described by its image $\sqrt{\mathbf{F}_q^2}$.

For $q \equiv 3 \pmod 4$ one can take the principal square root as a square-root function, i.e., take $\sqrt{\mathbf{F}_q^2} = \mathbf{F}_q^2$, as in Section 6.5.1; but the concept of principal square roots does not generalize to $q \equiv 1 \pmod 4$. For any odd prime q one can take

$\sqrt{\mathbf{F}_q^2} = \{0, 1, \ldots, (q-1)/2\}$. Other choices sometimes have computational benefits.

The definition of Elligator 2 is also parametrized by a non-square $u \in \mathbf{F}_q$. If $q \equiv 3 \pmod 4$ then one can take $u = -1$. If $q \equiv 5 \pmod 8$ then one can take $u = 2$. Finding a non-square is an easy computation in general since about half of the elements of $\mathbf{F}_q$ are non-squares. For efficiency it is desirable to choose u to be small, or otherwise to choose u to speed up multiplications by u.

6.7.2 The map

Theorem 12. *Let q be an odd prime power. Let A, B be elements of $\mathbf{F}_q$ such that $AB(A^2 - 4B) \neq 0$. Let u be a non-square in $\mathbf{F}_q$. Let $\sqrt{\ }$ be a square-root function for $\mathbf{F}_q$. Define R as the set*

$$\left\{r \in \mathbf{F}_q : 1 + ur^2 \neq 0,\ A^2ur^2 \neq B(1 + ur^2)^2\right\}.$$

The following elements of $\mathbf{F}_q$ are defined for each nonzero $r \in R$:

$$\begin{aligned} v &= -A/(1 + ur^2), \\ \epsilon &= \chi(v^3 + Av^2 + Bv), \\ x &= \epsilon v - (1 - \epsilon)A/2, \\ y &= -\epsilon\sqrt{x^3 + Ax^2 + Bx}\,. \end{aligned}$$

Furthermore $v\epsilon xy \neq 0$ and $y^2 = x^3 + Ax^2 + Bx$.

If $q \equiv 1 \pmod 4$ and $A^2 - 4B$ is a non-square in $\mathbf{F}_q$ then $R = \mathbf{F}_q$.

Proof. **v is defined and $v \neq 0$:** By hypothesis $A \neq 0$ and $1 + ur^2 \neq 0$.

$v^3 + Av^2 + Bv \neq 0$ and $\epsilon \neq 0$: Note that $v + vur^2 = -A$ so $v^2 + Av = v(v + A) = v(-vur^2)$. If $v^2 + Av + B = 0$ then $v^2ur^2 = B$ so, using the definition of v, $A^2ur^2 = B(1 + ur^2)^2$, contradicting the definition of R. Hence $v^2 + Av + B \neq 0$, so $v^3 + Av^2 + Bv \neq 0$, so $\epsilon \neq 0$.

$x^3 + Ax^2 + Bx$ is a nonzero square and $x \neq 0$: There are two cases. First case: $\epsilon = 1$, i.e., $v^3 + Av^2 + Bv$ is a nonzero square. Then $x = v$ so $x^3 + Ax^2 + Bx$ is a nonzero square; and $x \neq 0$ since $v \neq 0$.

Second case: $\epsilon = -1$, i.e., $v^3 + Av^2 + Bv$ is a non-square. Then $x = -v - A = vur^2$. All factors v, u, r^2 here are nonzero so $x \neq 0$; note also that $\chi(x) = \chi(v)\chi(u) = -\chi(v)$. Furthermore $x(x + A) = (-v - A)(-v) = v(v + A)$ so $x^2 + Ax + B = v^2 + Av + B$ so $\chi(x^3 + Ax^2 + Bx) = -\chi(v^3 + Av^2 + Bv) = -\epsilon = 1$ so $x^3 + Ax^2 + Bx$ is a nonzero square.

y is defined, $y^2 = x^3 + Ax^2 + Bx$, and $y \neq 0$: $x^3 + Ax^2 + Bx$ is a nonzero square so $\sqrt{x^3 + Ax^2 + Bx}$ is defined; all factors in $y = -\epsilon\sqrt{x^3 + Ax^2 + Bx}$ are nonzero, and $y^2 = x^3 + Ax^2 + Bx$.

If $q \equiv 1 \pmod 4$ and $A^2 - 4B$ is a non-square then $R = \mathbf{F}_q$: Fix $r \in \mathbf{F}_q$, and write $s = ur^2$. Then $\chi(s) \in \{0, -1\}$ while $\chi(\pm 1) = 1$ so $s \neq \pm 1$. In particular $1 + ur^2 \neq 0$.

Suppose that $A^2 s = B(1 + s)^2$. Subtract $4Bs$ to obtain $(A^2 - 4B)s = B(1 - s)^2$. Multiply to obtain $(A^2 - 4B)A^2 s^2 = B^2(1 + s)^2(1 - s)^2$. By hypothesis $\chi(A^2 - 4B) = -1$ so $\chi((A^2 - 4B)A^2 s^2) \in \{-1, 0\}$, while all of $B, 1 + s, 1 - s$ are nonzero so $\chi(B^2(1 + s)^2(1 - s)^2) = 1$, contradiction. Consequently $A^2 ur^2 \neq B(1 + ur^2)^2$. □

Definition 10. *In the situation of Theorem 12, the* **decoding function** *for the Weierstrass curve $E : y^2 = x^3 + Ax^2 + Bx$ is the function $\psi : R \to E(\mathbf{F}_q)$ defined as follows: $\psi(0) = (0, 0)$; if $r \neq 0$ then $\psi(r) = (x, y)$.*

With more work one can extend the definition of ψ to cover elements of $\mathbf{F}_q$ outside R. For example, if $A^2 - 4B$ is a square then one can map any square roots in $\mathbf{F}_q$ of $(A^2 - 2B \pm A\sqrt{A^2 - 4B})/(2uB)$ to $((-A \pm \sqrt{A^2 - 4B})/2, 0)$. However, our main interest is in the case $R = \mathbf{F}_q$, and then this extra work is unnecessary.

6.7.3 Inverting the map

Theorem 13. *In the situation of Definition 10:*

1. *If $r \in R$ then the set of preimages of $\psi(r)$ under ψ is $\{r, -r\}$.*
2. *$\psi(R)$ is the set of $(x, y) \in E(\mathbf{F}_q)$ such that*
   - *$x \neq -A$,*
   - *if $y = 0$ then $x = 0$, and*
   - *$-ux(x + A)$ is a square in $\mathbf{F}_q$.*
3. *If $(x, y) \in \psi(R)$ then the following element $\bar{r}$ of R is defined and $\psi(\bar{r}) = (x, y)$:*

$$\bar{r} = \begin{cases} \sqrt{-x/((x + A)u)} & \text{if } y \in \sqrt{\mathbf{F}_q^2}\,; \\ \sqrt{-(x + A)/(ux)} & \text{if } y \notin \sqrt{\mathbf{F}_q^2}\,. \end{cases}$$

Proof. **1.** If $r = 0$ then $r = -r$ so $\psi(r) = \psi(-r)$. If $r \neq 0$ then Theorem 12 defines $\psi(r)$ purely in terms of r^2 so $\psi(r) = \psi(-r)$.

Conversely, assume that $\psi(r) = \psi(r')$; our goal is to show that $r' \in \{r, -r\}$. If $r = 0$ then $\psi(r) = (0, 0)$, and otherwise $\psi(r)$ has nonzero coordinates by Theo-

rem 12; hence $r = 0$ if and only if $r' = 0$. The only remaining case is that $r \neq 0$ and $r' \neq 0$.

Define v', ϵ', x', y' from r' as in Theorem 12. Then $(x', y') = \psi(r') = \psi(r) = (x, y)$. Furthermore $y = -\epsilon\sqrt{x^3 + Ax^2 + Bx}$ and $y' = -\epsilon'\sqrt{(x')^3 + A(x')^2 + B(x')}$ so $\epsilon' = \epsilon$ since $\sqrt{\ }$ is a function. Next $x = \epsilon v - (1-\epsilon)A/2$ and $x' = \epsilon' v' - (1-\epsilon')A/2$ so $v' = v$. Finally $1 + ur^2 = 1 + u(r')^2$ so $r' \in \{r, -r\}$ as claimed.

2. Fix $r \in R$, and write $(x, y) = \psi(r)$. If $r = 0$ then $(x, y) = (0, 0)$ so $x = 0$; $x \neq -A$; and $-ux(x + A) = 0$ is a square. If $r \neq 0$ then (x, y) is defined in Theorem 12, and there are two cases. The first case is that $\epsilon = 1$; then $x = v \neq -A$ since $ur^2 \neq 0$. The second case is that $\epsilon = -1$; then $x = -v - A \neq -A$ since $v \neq 0$. In both cases $y \neq 0$ by Theorem 12, and $-ux(x + A) = -uv(v + A) = -uv(-vur^2) = u^2v^2r^2$, which is a square.

Conversely, assume that $(x, y) \in E(\mathbf{F}_q)$, that $x \neq -A$, then if $y = 0$ then $x = 0$, and that $-ux(x + A)$ is a square. Our goal is to show that $(x, y) \in \psi(R)$. We will in fact show more: $(x, y) = \psi(\bar{r})$ where $\bar{r}$ is defined as in the third part of the theorem statement.

If $y = 0$ then $x = 0$ by assumption. Furthermore $y = \sqrt{0} \in \sqrt{\mathbf{F}_q^2}$ and $-x/((x + A)u) = 0$ so $\bar{r}$ is defined as 0. Hence $(x, y) = (0, 0) = \psi(0) = \psi(\bar{r})$ as claimed.

Assume from now on that $y \neq 0$. The curve equation then implies that $x \neq 0$. Now $(x+A)u$ and ux are both nonzero, and both $-x/((x+A)u)$ and $-(x+A)/(ux)$ are squares, so $\bar{r}$ is defined and nonzero. We will see below that $1 + u\bar{r}^2 \neq 0$ and $A^2u\bar{r}^2 \neq B(1 + u\bar{r}^2)^2$, so $\bar{r} \in R$. Define $\bar{v}, \bar{\epsilon}, \bar{x}, \bar{y}$ as in Theorem 12.

If $y \in \sqrt{\mathbf{F}_q^2}$ then $\bar{r}^2 = -x/((x + A)u)$ so $1 + u\bar{r}^2 = A/(x + A)$ so $\bar{v} = -x - A$ so $\chi(\bar{v}) = \chi(-x - A) = \chi(ux) = -\chi(x)$. Next $\bar{v}^2 + A\bar{v} + B = x^2 + Ax + B$ so $\bar{\epsilon} = \chi(\bar{v}^3 + A\bar{v}^2 + B\bar{v}) = -\chi(x^3 + Ax^2 + Bx) = -1$. Consequently $\bar{x} = -\bar{v} - A = x$ and $\bar{y} = -\bar{\epsilon}\sqrt{x^3 + Ax^2 + Bx} = \sqrt{x^3 + Ax^2 + Bx} = y$.

The remaining case is that $y \notin \sqrt{\mathbf{F}_q^2}$, i.e., $y = -\sqrt{x^3 + Ax^2 + Bx}$. Then $\bar{r}^2 = -(x+A)/(ux)$ so $1+u\bar{r}^2 = -A/x$ so $\bar{v} = x$. Now $\bar{v}^3+A\bar{v}^2+B\bar{v} = x^3+Ax^2+Bx$ so $\bar{\epsilon} = 1$. Consequently $\bar{x} = \bar{v} = x$ and $\bar{y} = -\bar{\epsilon}\sqrt{x^3 + Ax^2 + Bx} = -\sqrt{x^3 + Ax^2 + Bx} = y$.

In both cases we have $\bar{v}^2 + A\bar{v} + B = x^2 + Ax + B \neq 0$ so $A^2u\bar{r}^2 \neq B(1 + u\bar{r}^2)^2$.

3. Fix $(x, y) \in \psi(R)$. We showed above that $x \neq -A$; that if $y = 0$ then $x = 0$; and that $-ux(x + A)$ is a square. We also showed that under these conditions $\bar{r}$ is defined and $\psi(\bar{r}) = (x, y)$. □

6.7.4 Encoding as strings

Theorem 14. *In the situation of Definition 10, assume that q is prime, that $q \equiv 1 \pmod 4$, and that $A^2 - 4B$ is not a square in $\mathbf{F}_q$. Define $b = \lfloor \log_2 q \rfloor$. Define $\sigma : \{0,1\}^b \to \mathbf{F}_q$ by $\sigma(\rho_0, \rho_1, \ldots, \rho_{b-1}) = \sum_i \rho_i 2^i$. Define $S = \sigma^{-1}(\{0, 1, 2, \ldots, (q-1)/2\})$. Define $\iota : S \to E(\mathbf{F}_q)$ as follows: $\iota(\rho) = \psi(\sigma(\rho))$. Then $\#S = (q+1)/2$; ι is an injective map from S to $E(\mathbf{F}_q)$; and $\iota(S) = \psi(\mathbf{F}_q)$.*

Proof. Note that $R = \mathbf{F}_q$ by Theorem 12; i.e., ψ is defined on all of $\mathbf{F}_q$. The rest of the proof is identical to the proof of Theorem 11 with ϕ replaced by ψ, τ replaced by ρ, Theorem 9 replaced by Theorem 12, and Theorem 10 replaced by Theorem 13. □

6.7.5 Application of ψ to Curve25519

Curve25519 is the curve $y^2 = x^3 + Ax^2 + Bx$ over $\mathbf{F}_q$ with $q = 2^{255} - 19$, $A = 486662$, and $B = 1$. Here $q \equiv 1 \pmod 4$ and $A^2 - 4B$ is a non-square in $\mathbf{F}_q$, so $R = \mathbf{F}_q$.

We take $u = 2$. We copy from [47] the following standard efficiently computable square-root function for $\mathbf{F}_q$. Given a square $a \in \mathbf{F}_q$ compute $b = a^{(q+3)/8}$; note that $q \equiv 5 \pmod 8$, so $(q+3)/8$ is an integer. Then $b^4 = a^2$, i.e., $b^2 \in \{a, -a\}$. Define $\sqrt{a}$ as $|b|$ if $b^2 = a$ and as $\left|b\sqrt{-1}\right|$ otherwise. Here $|b|$ means b if $b \in \{0, 1, \ldots, (q-1)/2\}$, otherwise $-b$.

Computing ι takes 1 square-root computation, 1 inversion, 1 computation of χ, and a few multiplications. Note that the inversion and the square-root computation can be combined into one exponentiation, as in [47]. The computation of χ can also be combined into the square-root computation as follows. First compute a power of $v^3 + Av^2 + Bv$ as above, obtaining a square root of $v^3 + Av^2 + Bv$ if $v^3 + Av^2 + Bv$ is a square. If the square of this power turns out to match $v^3 + Av^2 + Bv$ then $\epsilon = 1$ and $x = v$. Otherwise $\epsilon = -1$, $x = vur^2$, and $x^3 + Ax^2 + Bx = ur^2(v^3 + Av^2 + Bv)$; multiply the previous power by r and by a precomputed power of u to obtain a square root of $x^3 + Ax^2 + Bx$.

Similar comments apply to computing the inverse map. Checking for $P \in \iota(S)$ takes 1 squaring and a computation of χ to obtain $\chi((x + A/2)^2 - A^2/4) = \chi(x(x + A))$.

This map and its inverse seem simplest to describe in Weierstrass coordinates, but maps to and from Edwards form, and other curve shapes, are also easily obtained.

6.8 Conclusion

In this chapter we illustrated how widespread censorship is in today's communication, and we indicated some of the important measures taken to circumvent it. We discussed how public-key encryption, in particular elliptic-curve encryption, poses a security problem for protocols aiming at censorship circumvention. These protocols naturally send points on EC and these points are easy to distinguish.

To make these points indistinguishable from uniformly random strings we presented two maps: Elligator 1 and Elligator 2. They both efficiently map points on a single curve to bit-strings indistinguishable from random and back. About every second point of a curve can be mapped, but the tests to verify if the point can be mapped are easy.

Elligator 1 builds on the function from [133]. This map has a restricted number of curves that can be used with it, but does not exclude curves that are fast and secure according to the state-of-the-art criteria. This chapter presents the secure curve Curve1174 that was designed specifically for Elligator 1. Additionally, criteria to create new secure curves are discussed. Elligator 2 is comparable to Elligator 1 with respect to performance, but can work with wider range of curves, for example with Curve25519. Both maps presented in this chapter can be used with a wide range of existing ECC protocols.

Chapter 7

LightMix: mixing with minimal real-time asymmetric cryptographic operations

7.1 Introduction

Digital messaging has become a significant form of human communication, yet currently most of these systems do not provide basic protections of untraceability and unlinkability of messages. These protections are fundamental to freedom of inquiry, freedom of expression, and increasingly to online privacy. Grave threats to privacy exist from global adversaries who construct traffic-analysis graphs detailing who communicates with whom.

To provide anonymity online, a popular approach is onion routing, such as implemented in the widely used system ToR [111]. Onion-routing systems, however, have limitations on the level of anonymity achievable: most significantly, because they route different sessions of messages along different paths and they do not perform random permutations of messages, they are vulnerable to a variety of traffic-analysis attacks—for example [287, 99], as well as intersection attacks [52, 104, 101].

By contrast, mix-nets hold fundamentally greater promise to achieve higher levels of anonymity than do onion-routing systems because mix-nets are resilient against traffic-analysis attacks. Specifically, since all mix-net messages travel

through the same fixed cascade of mixnodes, observing the communication paths of messages within a mix-net is not useful to the adversary. Also, mix-nets can process larger batches of messages than can onion-routing systems, which is important because the batch size is the size of the anonymity set. Using a fixed cascade achieves resilience against intersection attacks [52].

The Internet of Things (IoT) is gradually becoming a part of everyday life. Evidence of this process includes the existing and projected explosive growth of mobile and light-weight devices connected to the Internet [273]. New IoT services often inherit approaches popular for modern Internet services, including centralized architectures, use of clouds, and identification in place of authentication. Together, these approaches facilitate over-collection of data [213] and extensive user profiling. Communication anonymity is a fundamental technology that can reduce privacy risks. It is also a building block that will support the development of other protection mechanisms, such as anonymous authentication within the IoT [6].

Mix-nets provide strong anonymity guarantees for their users. Adopting mix-nets in the IoT environment, however, is not straightforward due to the high computational costs imposed by current secure mix-net designs, due to their frequent use of asymmetric cryptographic operations on clients. Such expensive operations reduce battery life of a mobile device and reduce the computational capacity available for the device's main purpose. For example, implementing public-key encryption on passive RFID tags [15] will severely limit their capability to perform other operations.

In this chapter we introduce LightMix, a new approach to anonymous communications. LightMix is a new variant of fixed-cascade mixing networks (mix-nets). LightMix uses a precomputation phase to avoid all computationally-intensive public-key cryptographic operations in its core real-time protocol. Senders participate only in the real-time phase. Thus, senders never perform any public-key operations after an initial key-exchange during a one-time registration process. LightMix has drastically lower real-time cryptographic latency than do other mix-nets. Through its use of precomputation, and through its novel key management, LightMix is markedly different from all previous mix-nets. Due to its lack of public-key operations in its core real-time phase, it is well suited for applications running on computationally restricted devices in the IoT, as well as on mobile devices with restricted battery life, including smartphones.

In summary, in addition to its minimal use of real-time asymmetric cryptographic operations and its resistance to traffic analysis, LightMix enjoys each of the

following advantages: First, LightMix scales linearly (in number of users, number of mixnodes, and batch size). Second, unlike onion-routing and some mix-nets, its scalable design does not limit batch size, which means that LightMix can support large anonymity sets. Third, clients do not perform any complex public-key operations. Therefore, LightMix has low power consumption for clients, making it especially well-suited for applications on light-weight devices, including chat on smartphones.

Our main contributions are the design, preliminary analysis, and proof-of-concept implementation of LightMix, a new mix-net variant that, through precomputation, achieves lower real-time computational latency than do all existing mix-nets (traditional and re-encryption), while still benefiting from the strong anonymity properties of mix-nets over onion-routing systems.

In the rest of this paper we review related work, provide an overview of LightMix, describe the core LightMix protocol, explain some protocol enhancements, provide security arguments, compare performance of LightMix with that of other mix-nets, present timings of our proof-of-concept implementation, discuss several issues raised by LightMix, and present our conclusions.

7.2 System overview

Before defining LightMix's core protocol, we first explain our architecture and communication model, adversarial model, and security goals.

7.2.1 Architecture and communication model

LightMix is a new mix-net protocol that provides anonymous communication for its users (senders and receivers). The main goal is to ensure *unlinkability* of messages entering and leaving the system, though it is known which users are active at any given moment.

LightMix has n mixnodes that comprise a *fixed cascade*: all nodes are organized in a fixed order from the first node to the last. Within the LightMix system this order can be systematically changed and rotated, without affecting users in any way. Any message sent by a user is forwarded through all n servers. As with any mix-net, LightMix collects a certain number of messages in a batch before processing them. Section 7.9 discusses our strategy for assembling batches, though details may depend on the application.

To become a LightMix user, one must first establish for each mixnode a shared key. Section 7.3.2 provides more details on how these keys can be established. *Round keys* derived from the shared keys are used in each *round* of communication. A round begins with the start of batch processing.

For each round, β messages are collected and randomly ordered. Each message must have the same length, and all messages in a batch are processed simultaneously. The other messages are not accepted and are sent in a subsequent round. Every message is assigned to a corresponding incoming message *slot*, and is leaving mix node from an output slot. The relationship between an incoming slot and an output slot is defined by a random *permutation* selected by a mix node for the given round.

To process messages quickly in real time, LightMix performs precomputations that do not involve any user. The precomputations are performed in a separate phase during which LightMix executes all public-key encryptions, enabling the real-time computations to be carried using only fast multiplications.

7.2.2 Adversarial model

We assume authenticated communication channels between all mixnodes. Thus, an adversary can eavesdrop, forward, and delete messages between mixnodes, but not modify, replay, or inject new ones, without detection. For any communication not among mixnodes, we assume the adversary can eavesdrop, modify, or inject messages at any point of the network.

The goal of the adversary is to compromise the anonymity of the communication initiator, or to link inputs and outputs of the system. We consider applications where initiators are users of the LightMix system. We do not consider adversaries who aim to launch denial-of-service (DoS) attacks.

An adversary can also compromise users; however, we assume that at least two users are honest. Mixnodes can also be compromised, but at least one of them needs to be honest for the system to be secure. We assume compromised mixnodes to be malicious but cautious: they aim not to get caught violating the protocol.

We envision a deployment model in which there are dedicated trusted data centers serving as the mixnodes (perhaps competitively awarded). As such, they are incentivized not to be kicked out. By contrast, some mix-nets allow the mixnodes to enter and leave with low cost.

An implication of our deployment model is that it is sufficient to be able to detect a cheating node with sufficient probability at some point. By contrast, in more flexible deployment models, the nodes should prove more stringently that they have computed correctly before the output is opened. LightMix does require that the exit node commit to its output prior to being able to read the system output, which protects against certain adaptive attacks.

7.2.3 Security goals

LightMix aims to satisfy each of the following two security properties:

1. **Anonymity:** A protocol provides *anonymity* if the adversary cannot map any input message to the corresponding output message, with a probability significantly better than that of random guessing, even if the adversary compromises all but two users and all but one mixnode.

2. **Integrity:** A protocol provides *integrity* if at the end of every run involving β honest users, (a) either the β messages from the honest users are delivered unaltered to the intended recipients, or [(b) a malicious party is detected with a non-negligible probability and (c) no honest party is proven malicious].

7.3 The core protocol

We now present the core LightMix protocol, beginning with some preliminary notations and concepts, followed by a detailed specification.

7.3.1 Preliminaries

We introduce the primitives and notations used to describe the protocol. There are n mixnodes that process β messages per batch. For simplicity we assume here that the system already knows for each sender what position to use. When implementing the system this assignment can, for example, be achieved by including the sender's identity (possibly a pseudonym) when sending a message to the system.

All computations are performed in a prime-order cyclic group $\mathbf{G}$ satisfying the decision Diffie-Hellman (DDH) assumption. The order of the group is p, and g is a generator for this group. Let $\mathbf{G}^*$ be the set of non-identity elements of $\mathbf{G}$.

LightMix uses a multi-party group-homomorphic cryptographic system. We make use of a system based on ElGamal, described by Benaloh [39], though any such system could be used. This system works as follows:

- $e_i \in \mathbb{Z}_p^*$: the share for mixnode i of the secret key e.
- d: the public key of the system, based on the mixnode shares of the secret key: $d = \prod_i g^{e_i}$.
- $\mathcal{E}(m) = (g^x, m \cdot d^x)$, $x \in_R \mathbb{Z}_p^*$: encryption of message m under the system's public key d. We call g^x the *random component* and $m \cdot d^x$ the *message component* of the ciphertext. When applying encryption on a vector of values, each value in the vector is encrypted individually—each with a fresh random value—and the result is a vector of ciphertexts.
- $\mathcal{D}_i(g^x) = (g^x)^{-e_i}$: the decryption share for mixnode i computed from the random component of a ciphertext using the mixnode's share of the secret key. As with encryption, applying this function on a vector of random values results in a vector of corresponding decryption shares.

To decrypt a ciphertext $(g^x, m \cdot d^x)$, all parties need to cooperate because the decryption shares for all mixnodes are required to retrieve the original message:

$$\begin{aligned} & m \cdot d^x \cdot \textstyle\prod_{i=1}^n \mathcal{D}_i(g^x) \\ = \quad & m \cdot (\textstyle\prod_{i=1}^n g^{e_i})^x \cdot \textstyle\prod_{i=1}^n (g^x)^{-e_i} \\ = \quad & m\,. \end{aligned}$$

The LightMix protocol uses the following values:

- $r_{i,a}, s_{i,a} \in \mathbf{G}^*$: random values (freshly generated for each round) of mixnode i for slot a. Thus, $\mathbf{r}_i = (r_{i,1}, r_{i,2}, \ldots, r_{i,\beta})$ is a vector of random values for the β slots in the message map at mixnode i. Similarly, $\mathbf{s}_i$ is also a vector of random values for mixnode i.
- π_i: a random permutation of the β slots used by i. The inverse of the permutation is denoted by π_i^{-1}.
- $k_{i,j} \in \mathbf{G}^*$: a group element shared between mixnode i and the sending user for slot j. These values are used as keys to blind messages.
- $M_j \in \mathbf{G}^*$: the message sent by user j. Like other values in the system, these values are group elements. They can be easily converted from, for example,

an ASCII-encoded string. The group size determines the length of an individual message that can be sent.

For readability we introduce the following shorthand notations:

- $\mathbf{R}_i$: the direct product of all local random $\mathbf{r}$ values through mixnode i; i.e., $\mathbf{R}_i = \prod_{j=1}^{i} \mathbf{r}_j$.
- $\mathbf{S}_i$: the product and permutation of all local random s values:

$$\mathbf{S}_i = \begin{cases} \mathbf{s}_1 & i = 1 \\ \pi_i(\mathbf{S}_{i-1}) \times \mathbf{s}_i & 1 < i \leq n\,. \end{cases}$$

- $\mathbf{\Pi}_i(a)$: the permutation performed by LightMix through mixnode i, i.e., the composition of all individual permutations:

$$\mathbf{\Pi}_i(a) = \begin{cases} \pi_1(a) & i = 1 \\ \pi_i(\mathbf{\Pi}_{i-1}(a)) & 1 < i \leq n\,. \end{cases}$$

- $\mathbf{k}_i$ and $\mathbf{k}_i^{-1}$: the vector of keys shared between mixnode i and the users for all β slots and their inverses, respectively; $\mathbf{k}_i = (k_{i,1}, k_{i,2}, \ldots, k_{i,\beta})$ and $\mathbf{k}_i^{-1} = (k_{i,1}^{-1}, k_{i,2}^{-1}, \ldots, k_{i,\beta}^{-1})$.
- K_j: the product of all shared keys of the sending user for slot j: $K_j = \prod_{i=1}^{n} k_{i,j}$.
- $\mathbf{K}$ is a vector of products of shared keys for the β slots; $\mathbf{K} = (K_1, K_2, \ldots, K_\beta)$.

7.3.2 Protocol description

We now present the core protocol. In this explanation we focus on simplicity and clarity; see Sections 7.4 and 7.6 for a discussion of possible security issues and enhancements. We separately discuss each of the three protocol phases: setup, precomputation, and real time.

Setup

In the setup phase, the mixenodes establish their secret shares e_i and the shared public key d, which are used for the multi-party homomorphic encryption scheme.

The users also establish their keys $k_{i,j}$, which they share with all mixenodes. This can be done using any (offline) key distribution method. One way to derive

these keys is using a Diffie-Hellman key exchange. The resulting key can be used as a seed to derive unique values for $k_{i,j}$ for every session. Depending on the chosen key distribution protocol, this would be the only time a user is possibly required to perform an asymmetric cryptographic operation. The key exchange must be performed once for each user, and this exchange can be carried during the user's enrollment into the system.

Precomputation

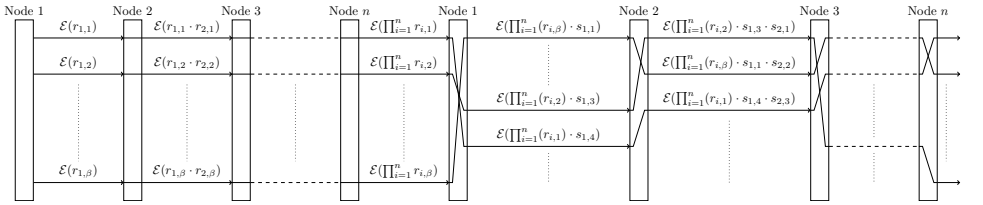

Figure 7.1: A schematic example of the first two steps of the precomputation phase, which result in the values $\mathcal{E}(\mathbf{\Pi}_n(\mathbf{R}_n) \times \mathbf{S}_n)$.

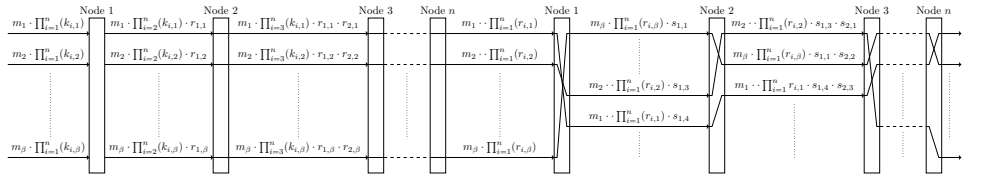

Figure 7.2: A schematic example of the first two steps of the real-time phase, which result in the values $\mathbf{\Pi}_n(\mathbf{M} \times \mathbf{R}_n) \times \mathbf{S}_n$.

The precomputation phase is performed only by the mixenodes, without any involvement from the users. It is performed once for each real-time phase. Shared values are established to circumvent the need for public-key operations during the real-time phase. The precomputation phase comprises three different steps given below. The goal of the precomputation phase is to compute the values $\mathbf{\Pi}_n(\mathbf{R}_n) \times \mathbf{S}_n$, which are used in the real-time phase. Figure 7.1 shows a schematic example of the first two steps of the precomputation phase.

Step 1 - Preprocessing. The mixnodes start by generating fresh **r**, **s**, $\boldsymbol{\pi}$ values. Then they collectively compute the product of all of their individual **r** values under encryption using the public key d of the system, which was computed during the setup

phase. This computation takes place by each mixnode i sending the following message to the next mixnode:

$$\mathcal{E}(\mathbf{R}_i) = \begin{cases} \mathcal{E}(\mathbf{r}_1) & i = 1 \\ \mathcal{E}(\mathbf{R}_{i-1}) \times \mathcal{E}(\mathbf{r}_i) & 1 < i \leq n\,. \end{cases}$$

Each mixnode encrypts its own **r** values and uses the homomorphic property of the encryption system to compute the multiplication of this ciphertext with the input it receives from the previous mixnode. Eventually, the last mixnode sends the final values $\mathcal{E}(\mathbf{R}_n)$ to the first mixnode as input for the next step.

Step 2 - Mixing. In the second step, the mixnodes together mix the values and compute the results $\mathbf{\Pi}_n(\mathbf{R}_n) \times \mathbf{S}_n$, under encryption. This is accomplished having every mixnode i send the following message to the next mixnode:

$$\begin{aligned} &\mathcal{E}(\mathbf{\Pi}_i(\mathbf{R}_n) \times \mathbf{S}_i) \\ &= \begin{cases} \pi_1(\mathcal{E}(\mathbf{R}_n)) \times \mathcal{E}(\mathbf{s}_1) & i = 1 \\ \pi_i(\mathcal{E}(\mathbf{\Pi}_{i-1}(\mathbf{R}_n) \times \mathbf{S}_{i-1})) \times \mathcal{E}(\mathbf{s}_i) & 1 < i \leq n\,. \end{cases} \end{aligned}$$

As with the first step, the last mixnode sends the final encrypted values $\mathcal{E}(\mathbf{\Pi}_n(\mathbf{R}_n) \times \mathbf{S}_n)$ to the first mixnode. These final values now must be decrypted together by all mixnodes, which happens in the last step of the precomputation.

Step 3 - Postprocessing. To complete the precomputation, the mixnodes decrypt the precomputed values. Each mixnode i computes its decryption shares $\mathcal{D}_i(\mathbf{x})$, where $(\mathbf{x}, \mathbf{c}) = \mathcal{E}(\mathbf{\Pi}_n(\mathbf{R}_n) \times \mathbf{S}_n))$. The message parts **c** are multiplied with all the decryption shares to retrieve the plaintext values $\mathbf{\Pi}_n(\mathbf{R}_n) \times \mathbf{S}_n$. This computation can be carried out either using another pass through the network (in which every mixnode multiplies in its own decryption share), or by having all mixnodes send their encryption shares to the last mixnode, which can then perform the multiplication. The last mixnode to be used in the real-time phase stores the decrypted precomputed values.

Real time

For the real-time phase, each user constructs its input by taking its message m and multiplying it with its combined shared key k to compute the blinded message $m \times k$. This blinded message is then sent to the mix-net. One option would be to send the blinded messages to the first mixnode. Once enough blinded messages

are received, they are combined to yield the vector $\mathbf{M} \times \mathbf{K}$. As in the precomputation phase, the real-time phase can again be split into three steps. Figure 7.2 gives a schematic example of the first two steps.

Step 1 - Preprocessing. During the preprocessing step, the mixnodes take out the keys **k** they share with the users and add in their **r** values to blind the original messages. This computation is performed by each mixnode i sending the following to the next mixnode:

$$\mathbf{M} \times \mathbf{K} \times (\prod_{j=1}^{i} \mathbf{k}_j^{-1} \times \mathbf{r}_j) = \mathbf{M} \times \mathbf{K} \times (\prod_{j=1}^{i-1} \mathbf{k}_j^{-1} \times \mathbf{r}_j) \times \mathbf{k}_i^{-1} \times \mathbf{r}_i \,.$$

The last mixnode sends the final values $\mathbf{M} \times \mathbf{R}_n = \mathbf{M} \times \mathbf{K} \times \prod_{j=1}^{n} \mathbf{k}_i^{-1} \times \mathbf{r}_i$, which are the blinded versions of the original messages, to the first mixnode as input for the next step. Now the user-specific keys **k** are taken out and replaced by the user-independent values **r**.

Step 2 - Mixing. The second step performs the mixing to hide the association between sender and receiver. The **s** values are added in to hide which input message corresponds to which output message. Each mixnode i (except the last mixnode) sends the following message to the next mixnode:

$$\begin{aligned} &\mathbf{\Pi}_i(\mathbf{M} \times \mathbf{R}_n) \times \mathbf{S}_i \\ &= \begin{cases} \pi_1(\mathbf{M} \times \mathbf{R}_n) \times \mathbf{s}_1 & i = 1 \\ \pi_i(\mathbf{\Pi}_{i-1}(\mathbf{M} \times \mathbf{R}_n) \times \mathbf{S}_{i-1}) \times \mathbf{s}_i & 1 < i < n \,. \end{cases} \end{aligned}$$

Finally, the last mixnode computes:

$$\mathbf{\Pi}_n(\mathbf{M} \times \mathbf{R}_n) \times \mathbf{S}_n = \pi_n(\mathbf{\Pi}_{n-1}(\mathbf{M} \times \mathbf{R}_n) \times \mathbf{S}_{n-1}) \times \mathbf{s}_n \,.$$

Now every mixnode performed its mixing, destroying the associations between senders and receivers. The last step retrieves the permuted original messages.

Step 3 - Postprocessing. The last mixnode can perform the final step. This mixnode retrieves the locally stored precomputed values $\mathbf{\Pi}_n(\mathbf{R}_n) \times \mathbf{S}_n$. To retrieve the permuted messages it now needs only to perform the following computation, using the result from the previous mixing step:

$$\mathbf{\Pi}_n(\mathbf{M}) = \mathbf{\Pi}_n(\mathbf{M} \times \mathbf{R}_n) \times \mathbf{S}_n \times (\mathbf{\Pi}_n(\mathbf{R}_n) \times \mathbf{S}_n)^{-1} \,.$$

How the messages are then delivered to the recipients depends on the application and is independent from LightMix. This step concludes the real-time phase, in which no public-key operations are performed.

7.4 Protocol integrity

The cryptographic construction presented in Section 7.3 expects the mixnodes and users to be honest but curious (they follow the protocol but may try to learn as much as possible). As for most mix-net protocols [80, 242, 183, 250], LightMix requires additional measures to ensure that mixnodes cannot tamper with the messages nor with the flow without detection. In this section, we augment the protocol to protect its anonymity and integrity against malicious attacks by users and by compromised mixnodes, which we shall call "adversarial mixnodes."

The overall strategy relies on the following assumption: We assume that compromised mixnodes are malicious but cautious. Each user periodically sends "trap" dummy messages. After the round completes, the user requests to open the path for the slot in which she sent dummy messages. In response, all messages exchanged and values used in this path during the real-time and precomputation phases are verified. This additional security mechanism is practical, and its cost and complexity is similar to that required for any mix-net.

7.4.1 Integrity of values and messages

To enable honest mixnodes to verifiably detect any malicious mixnodes that employ incorrect values or permutations in the precomputation or real-time phase, LightMix augments communications with proofs. To this end, all messages exchanged between mixnodes are signed using a digital signature scheme with existential unforgeability under an adaptive chosen-message attack [154]. In addition, during precomputation, each mixnode commits to the permutation π_j it applies to the incoming slot j using a perfectly hiding commitment (Commit) scheme [161] and signs that commitment. All the nodes broadcast their signed commitment using a reliable broadcast (Broadcast) protocol [115] [293]. Doing so makes it possible to reconstruct and verify all individual values $r_{i,j}$, $s_{i,j}$ and $\pi_{i,j}$ that mixnode j applies to slot i.

In the real-time phase, the users become involved, making the process more complicated because they do not perform any public-key operations during the real-time phase. The values that we need to verify that depend on the users are the $k_{i,j}$ values (shared between a mixnode i and the user in slot j), and the $m_j \cdot K_j$ values (the blinded message that a user sends in slot j).

Because the k values are shared between a mixnode and a user, we need them to agree on the commitments to these values. If we detect an anomaly involving

the k values, and a mixnode and user disagree on the value used, the commitment needs to provide proof of who misbehaved.

For this task, the following procedure can be followed in case the k values are derived from a seed that is agreed upon by a mixnode and user during the setup phase. After establishing this seed, the mixnode will compute a commitment of the seed and provide this commitment to the user. The mixnode generates the commitment in a way that enables the user to reveal the commitment and prove to other parties that the mixnode generated the commitment. The user must verify the commitment during the setup phase. This verification requires an additional public-key operation, though it will be performed only once provided the protocol runs without any disturbance.

Message integrity at entry

Messages at entry to LightMix need to arrive at LightMix and pass through the non-permuted part of the protocol without any undetected modification. Providing integrity of messages at this point is a challenge if one wishes to keep clients free from using any asymmetric cryptography during the real-time phase. We propose a construction where message authentication codes (MACs) are generated over the input message to the LightMix system.

To accomplish this goal, we introduce additional key values $l_{i,j}$, shared between mixnode i and the user for slot j, established and committed to in a similar way as for the k values. When the user sends the blinded message $m_j \cdot K_j$ to the system, the user also sends the following messages:

$$h_j = \mathsf{Hash}(m_j \cdot K_j) \text{ and } (MAC_{l_{1,j}}(h_j), \ldots, MAC_{l_{n,j}}(h_j)) \,.$$

During the real-time phase, the first mixnode starts by checking its corresponding MAC value in the list. If it is incorrect, the mixnode informs the other mixnodes and does not forward the computed value for this slot. If the MAC value is correct, it forwards the h values and the MAC values for the other mixnodes, together with its basic computed values.

Each subsequent mixnode follows the same procedure. At the end of the first step of the real-time phase, all mixnodes have checked their MAC values on the received h values, or the value is not processed any further. It is hard to enforce that the h_j value is related to the actual message that is being modified and forwarded further during Step 1 of the real-time phase. This value, however, is verified during

trap opening. Section 7.4.3 provides more details on how to pinpoint the misbehaving party.

7.4.2 Message tagging detection

A message-tagging attack is an attack where the adversary can mark a message at some point during the process, such that it is recognizable when it is output, compromising unlinkability between the inputs and outputs [268]. To perform a tagging attack unnoticed, the tag should also be removed before the messages are output. Tagging attacks are a threat to all mix-nets that use some form of malleable encryption, such as homomorphic encryption or group multiplications, where valid messages can be recognized when output by the mix-net. For example, Pfitzmann [262] presents such an attack on re-encryption mix-nets.

A simple example of a tagging attack is the following: The last mixnode multiplies the blinded message in one of the slots j with an additional factor t in the first step of the real-time phase. Now the blinded message in slot j will be $m_j \cdot \prod_{i=1}^{n} r_{i,j} \cdot t$ at the end of the first step, whereas the values in the other slots stay the same. When the last mixnode performs Step 3, it will see the final messages before it outputs them. If the messages are recognizable as valid outputs, it will observe that one of the messages does not seem to be valid. If this invalid message becomes valid when multiplied with t^{-1}, this message is likely the tagged one. The mixnode can now link the message, and possibly the recipient, to the sender. The mixnode can remove the tag and output the original messages, making it unobservable for the users and other mixnodes that a tagging attack took place.

Detection. To protect against these kinds of attacks and make them detectable, only small changes are needed to the protocol:

- **Precomputation Phase - Step 3:** The mixnodes no longer send out their decryption shares to retrieve the precomputed values. Instead, they keep their shares secret and publish only a commitment to them. The last mixnode also publishes a commitment to the message component of the ciphertext. The commitments can be computed, for example, using only one signature per mixnode for the decryption shares for all slots simultaneously. The plaintext results of the precomputation phase are thus no longer retrieved in that phase itself.
- **Real-Time Phase - Step 3:** The output of the mixing step $\mathbf{\Pi}_n(\mathbf{M} \times \mathbf{R}_n) \times \mathbf{S}_n$ is published by the last mixnode. Afterwards, all mixnodes release their decryp-

tion shares $\mathcal{D}_i(\mathbf{x})$ and the message component of the ciphertext $\mathbf{c}$. The output messages are then computed as follows, where $(\mathbf{x}, \mathbf{c}) = \mathcal{E}(\mathbf{\Pi}_n(\mathbf{R}_n) \times \mathbf{S}_n)$:

$$\mathbf{\Pi}_n(\mathbf{M} \times \mathbf{R}_n) \times \mathbf{S}_n \times \mathbf{c} \times \prod_{i=1}^{n} \mathcal{D}_i(\mathbf{x}) \,.$$

Because the mixnodes committed to all of the values necessary to retrieve the precomputed value, they cannot change these values to take out a possible tag anymore. The output of the mixing step does not reveal anything yet about the messages, because the r and s values are still included. The precomputed value to take out these values is retrieved only after the output of the mixing step is made public. Therefore, the last mixnode would not know from which output slot a possible tag should be removed before the output of the mixing step is made public. Once the output is made public, the tag cannot be removed because all computations for the third step are fixed and can be verified by anyone. This technique is helpful to detect the malicious party once a trap is opened (see Section 7.4.3).

7.4.3 Sending and verifying trap messages

To ensure the protocol is functioning correctly, we let users send trap messages with a certain probability, and request to open message paths of these traps. Opening of a path includes verification of all messages exchanged between mixnodes, the incoming message from the user, as well as intermediate values and permutations.

Sending trap messages and requesting to open them

To send a trap, a user starts by forming a message with a round ID, user ID, and statement that this is the dummy message. She calculates a MAC of this message with every key $l_{i,j}$ shared between mixnode i and the user for slot j. The trap is then the message together with its MAC values. It is encrypted according to the protocol and sent to the entry mixnode. After the round is over, the trap message appears at the output. Nodes verify the correctness of the round ID and MAC values. If the round ID or at least one of the MAC values is incorrect, they do not proceed. Otherwise, they send a user an authenticated message notifying her that the dummy is received, including an incremented counter of all dummies received from this user.

If a user does not receive any notification message, because a dummy was tagged or because one of the MAC values was not accepted by the disputing mixnode, she initiates a verification procedure. She sends a request to open the path

with her dummy message directly to all mixnodes, again signed by the MAC. If there is a mixnode disputing the validity of the MAC intended for him, and if the mixnode and user cannot agree on the key $l_{i,j}$ that is supposed to be used, the commitment to the seed of key $l_{i,j}$ (made by the mixnode and verified by the user during setup) is opened.

This process allows the key value to be retrieved, and the correctness of the MAC signed by this key is verified. If the retrieved key value does not correspond to what the mixnode claims, it must either agree to open the path, or be considered malicious. Otherwise if the key does not correspond to what the user claims, the user's request is dropped. If the user still wants to insist on verification, it has an option to sign the request with her private key.

Once it is established that the given user in the given round sent a trap, the user receives a notification, if the user has not received one before. Then mixnodes start opening the path in the precomputation phase of the message slot j in which the user sent his trap in the particular round.

Path opening for the precomputation phase

For the input slot j that we want to verify, the mixnodes have to decrypt exchanged messages for this slot and compute the r values in the non-permuted part. For the permuted part, mixnodes subsequently reveal the corresponding permutations and decrypt exchanged messages to obtain the s values. Doing so we can follow the computation through the mix-net and verify whether the precomputed value that was output was correctly computed.

To check, for example, whether mixnode i performed its computation correctly, that mixnode needs to present the signature from the previous mixnode on the values it received. The next mixnode will also have to present the signature it received from mixnode i to obtain proof what values were output by mixnode i. Once all information about the input, output, and values used in the expected computation are known, due to the signatures, commitments, and threshold decryption, it can be verified whether mixnode i performed the computation as expected.

Path opening for the real-time phase

Malicious mixnodes can also employ incorrect messages, values, and permutations in the real-time phase. We wish to make the mixnodes accountable for their be-

havior. One challenge is that malicious users may try to victimize some honest mixnodes by providing inconsistent inputs and later deny having done so.

First, the mixing step is verified. This check is performed in a similar fashion as for the precomputation phase. For the preprocessing step, a comparable approach is followed. To verify whether the correct input from the user to the system is used, all the keys l for the MAC values for the suspicious slot are output. The purpose is to detect if a mixnode or user changed the input. Because the values were processed in the mixing step, during the first step of the real-time phase, all mixnodes accepted the MAC values and thus should be able to provide a correct key. The mixnodes also reveal their r and k values for the corresponding slot, and the mixnodes check whether they performed their computations correctly. Although the mixnodes committed to the r values, they have not committed to individual k values. Therefore the sender must be involved in this process. The sender will also release all the k values it used for this message.

If a k value released by the sender does not match the one released by the corresponding mixnode, either the mixnode or the sender is misbehaving. The sender might do this to blame a mixnode of misbehaving to cause it to be removed from the system. This dispute needs to be resolved by having the user reveal the commitment by the mixnode on the shared keys. Doing so might also reveal k values used in previous sessions, but these values are only one of the components of K and therefore do not leak the original messages. There are two possibilities: either the user was malicious, in which case we do not care about his or her previous messages, or the mixnode was malicious, in which case we consider the k values to be compromised already. This procedure will reveal who acted maliciously and modified the output message.

7.4.4 Integrity analysis

We shall argue informally that, if any of the messages are altered by any node, (a) no honest party can be proven malicious, and (b) at least one malicious party is detected with non-negligible probability.

No honest party is proven malicious An honest mixnode can be targeted by some malicious colluding users or mixnodes that provide inconsistent inputs. If the output is corrupted by $n - 1$ colluding mixnodes that provide incorrect decryption shares, then validating the commitments will reveal this fact. Because the commit-

ment scheme is secure, malicious mixnodes cannot publish incorrect commitments without detection.

If the previous mixnode has provided inconsistent input, then the current mixnode can prove its correctness by providing the input signed by the previous mixnode, since all messages exchanged between mixnodes are digitally signed. If a mixnode falsely claims that the previous mixnode has provided incorrect input, it will have to provide the input signed by the previous node as proof. In both cases, the adversary will be successful only if the digital signature scheme is not secure.

If the user has provided inconsistent input and claimed falsely that the mixnode is malicious, then the mixnode can prove its innocence using the MAC value received from the mixnode. If the user and the mixnodes show disagreement about the key used for computing the MAC value, the seed for generating the key is revealed. In the setup phase, the mixnode generates a commitment to the seed, and the user verifies the commitment. Thus, the user can successfully claim a different seed for that commitment only when the commitment scheme is not secure.

At least one malicious party is detected with non-negligible probability We assume our adversary is malicious but cautious. So, if there is even a slight chance of a malicious node being detected and thrown out of the system, it will refrain from engaging in any such mischief. We now show that if an adversary behaves maliciously, there is a significant probability that it will be caught.

Let p denote the probability with which a user will send a trap message. In each round there are a total of β messages. The expected number of trap messages in a batch is $\alpha = p\beta$.

We want to calculate the probability that the adversary will be detected even if it modifies only one path. When paths are opened, the probability that the corrupted path will be chosen among one of the α trap paths is $\binom{\beta-1}{\alpha-1}/\binom{\beta}{\alpha} = \alpha/\beta = p$.

Thus, in case of malicious behavior, at least one of the adversarial nodes will be caught with probability p. If $p = 1/2$, we can use 2β slots to ensure an anonymity set of β, while having a significant probability to detect a malicious node.

7.5 Anonymity analysis

We analyze the anonymity of the LightMix protocol. First, we state a precise definition for a desired anonymity property. Second, we sketch a proof that LightMix

satisfies this definition. Third, we explain why LightMix resists well-known attacks on mix-nets.

7.5.1 Formal anonymity analysis

Motivated by [300, 64], we define our desired anonymity property in the form of a game between an adversary $\mathcal{A}$ and a challenger $\mathcal{C}$, where the adversary $\mathcal{A}$ is a probabilistic polynomial time (PPT) Turing machine. Upon request, $\mathcal{C}$ runs the protocol and returns the outputs to $\mathcal{A}$. For ease of exposition, we fix the number of users to β and assume each user sends a message during every round. Our static adversary $\mathcal{A}$ is allowed to compromise all but two users, as well as all but one mixnode before execution begins. The challenger $\mathcal{C}$ executes the protocol on behalf of honest nodes and users, while $\mathcal{A}$ acts on behalf of the compromised users and nodes. The adversary can eavesdrop on all messages among the mixnodes, but it cannot modify them; thus, $\mathcal{C}$ also sends a copy of every communication between two honest parties along with its source and destination to $\mathcal{A}$. We denote the output of the adversary from this game as $\langle \mathcal{A} | \mathcal{C} \rangle$.

The anonymity game works as follows:

Setup phase: The challenger $\mathcal{C}$ runs the setup for all honest nodes, and provides all public information to the adversary $\mathcal{A}$.

Query phase: As many times as $\mathcal{A}$ requests, $\mathcal{C}$ takes input messages in the form of (sender, message)-pairs for all the β slots from $\mathcal{A}$, and runs the LightMix protocol with those inputs.

Challenge phase: $\mathcal{A}$ chooses two honest users S_0 and S_1 and two distinct messages m'_0 and m'_1. $\mathcal{A}$ also chooses messages for all other honest users as m, where $m \notin \{m'_0, m'_1\}$, and sends this challenge to $\mathcal{C}$. The challenger $\mathcal{C}$ tosses a uniform random coin to obtain a bit b, and assigns $m_0 = m'_b$ and $m_1 = m'_{1-b}$, the messages corresponding to users S_0 and S_1, respectively. The challenger $\mathcal{C}$ then runs the protocol with these input messages, and gives the output message set to the adversary.

Query phase: After the challenge run, as many times as $\mathcal{A}$ requests, $\mathcal{C}$ takes message inputs from $\mathcal{A}$ and runs the LightMix protocol again.

Output phase: Finally, the adversary outputs $\overline{b}$ as its guess for b.

The adversary's advantage in the anonymity game is equal to $Pr[0 = \langle A|C\rangle \mid b = 0] - Pr[1 = \langle A|C\rangle \mid b = 0]$. We now define the anonymity property as follows:

Definition 1. *A protocol maintains* anonymity *if the advantage of the adversary in the anonymity game is negligible.*

The following theorem states that the LightMix protocol satisfies Definition 1.

Theorem 15. *If $\mathcal{E}$ is a CPA-secure group-homomorphic encryption system, and* Commit *is a perfectly-hiding non-interactive commitment scheme, then LightMix offers anonymity as defined in Definition 1 in the random oracle model.*

Sketch. We prove the security of LightMix by reduction from the security of the encryption system $\mathcal{E}$. In our argument, we use an equivalent modified version of the standard encryption game used to define the security of $\mathcal{E}$: on challenge message pair (m_0, m_1), the challenger $\mathcal{C}_\mathcal{E}$ of the encryption system returns a ciphertext pair $(\mathcal{E}(m_b), \mathcal{E}(m_{1-b}))$ instead of just one ciphertext $\mathcal{E}(m_b)$.[1]

Without loss of generality, we assume that $\mathcal{A}$ can compromise $\beta - 2$ users and $n - 1$ nodes. Let S_0 and S_1 be any two honest users and let i be the only honest mixnode in the system.

During the setup phase, the challenger $\mathcal{C}$ initiates the CPA game with the encryption challenger $\mathcal{C}_\mathcal{E}$, and ensures that the encyption public key in LightMix is the same as the public key in the CPA-security game with $\mathcal{C}_\mathcal{E}$. In the query phase, $\mathcal{C}$ can easily simulate the honest nodes, including the decryption shares for those nodes. It can do so because it manages and checks the hashing (random) oracle queries made by the adversary for its commitments to messages as well as permutations, and since it can open the committed decryption shares (obtained using the perfectly-hiding commitment scheme) with its own choice of values.

In the challenge phase, $\mathcal{C}$ assigns the corresponding messages $m_0 = m'_{\hat{b}}$ and $m_1 = m'_{1-\hat{b}}$, as determined by a random bit $\hat{b}$. In addition, it chooses s values (say $s_{i,0}$ and $s_{i,1}$) for the slots of S_0 and S_1 at honest node i such that $m_0/m_1 = s_{i,0}/s_{i,1}$. The challenger $\mathcal{C}$ then runs the challenge phase with the CPA-security challenger $\mathcal{C}_\mathcal{E}$ with $s_{i,0}$ and $s_{i,1}$ as two challenge messages. $\mathcal{C}_\mathcal{E}$ returns with a ciphertext pair $\mathcal{E}(s_{i,b})$ and $\mathcal{E}(s_{i,1-b})$. The anonymity-game challenger $\mathcal{C}$ uses this response pair in the precomputation phase for node i and slots S_0 and S_1. This way the challenger $\mathcal{C}$ embeds its decryption challenge from the CPA-security game into an unknown permutation for an honest player in the anonymity game of LightMix.

[1] By defining an appropriate hybrid in the simulation, this modified game can be easily shown equivalent to the standard CPA-security encryption game.

In the real-time phase of LightMix, the challenger $\mathcal{C}$ runs the protocol in a standard manner, except at the mixing step for node i. Here, we use $s_{i,1}$ and $s_{i,0}$ for slots S_0 and S_1, resepctively, such that $m_0 s_{i,1} = m_1 s_{i,0}$. Doing so allows $\mathcal{C}$ to open any of two honest real-time messages to any of the m_0 and m_1 values.

Finally, during the postprocessing step of the real-time phase, the challenger $\mathcal{C}$ knows all permutations except for two slots associated with m_0 and m_1 (i.e., except for $m'_{\hat{b}}$ and $m'_{1-\hat{b}}$). As a result, it can open the decryption shares appropriately for all other messages. Since $m_0 s_{i,1} = m_1 s_{i,0}$, the challenger can open m_0 and m_1 in any order.

In the end, if the anonymity game adversary A predicts the bit $\hat{b}$ correctly, the challenger $\mathcal{C}$ knows that its choice of $s_{i,1}$ and $s_{i,0}$ for slots S_0 and S_1 in the real-time phase matched the challenge ciphertexts used in the precomputation phase. In this case, it outputs $b = 1$ to the encryption game challenger $\mathcal{C}_{\mathcal{E}}$. Otherwise, if the adversary guesses $\hat{b}$ incorrectly, the challenger $\mathcal{C}$ outputs 0 to $\mathcal{C}_{\mathcal{E}}$.

As a result, if the LightMix protocol does not provide anonymity, we can apply it to break the CPA-security of the underlying group-homomorphic encryption system, a contradiction. □

7.5.2 LightMix resists standard mix-net attacks

We explain how LightMix resists standard attacks on mix-nets. There are several active attacks against fixed-cascade mix-nets. The literature can be confusing because there is no common nomenclature for these attacks, so we define each attack type we discuss.

Many of these attacks are based on adding, deleting, or modifying messages in the mix-net at the entrance. An adversary can either block incoming messages from $\beta - 1$ users, or replace them from the batch with her own, allowing her to trace a target message. Such attacks are called $n - 1$ or *flooding* attacks [80].

Replay attacks [80] [52] work by retransmitting a message from a previous session. These attacks can work only if encryption or re-encryption is deterministic. A generalization of these attacks is called a *blending* attack [114] and happens when the adversary manipulates messages that are in the same batch that includes the target messages. Dingeldine et al. [114] discuss a number of countermeasures. Section 7.4.2 addresses tagging attacks and our protections against them. Because the **K**, **R**, and **S** values are never reused, LightMix protects against *replay attacks* (see Section 7.4).

Contextual attacks [268], sometimes referred as *traffic-confirmation attacks* [269], *communication pattern attacks* [268], and intersection attacks [51], analyze the time when particular senders and receivers participate in the protocol, their communication patterns, and how many messages they send and receive. Only unobservable [260] systems protect against this type of attack. These attacks are sometimes included in the following category of attacks.

Intersection attacks and *statistical disclosure attacks* [52, 104, 101] make use of mix-net topologies that allow users to choose routes freely for their messages (*free mix routes*). In such systems, sets of messages in a batch of a mixnode can be distinguished, for example, since they come from different mixnodes or have different route lengths. Assuming that users often use the same routes for their messages, these routes can be distinguished by analyzing network flow data. Because LightMix uses a fixed cascade of mixnodes [52], LightMix is not susceptible to this family of attacks.

Traffic-analysis attacks are targeted at connection-based anonymity systems, as opposed to message-based systems. These connection-based systems often do not batch and permute incoming packets, and they use free mix routes. This approach permits an adversary to distinguish these paths based on measures such as counting packets [287] and timing communications [99]. These attacks do not work on LightMix because LightMix permutes messages in batches using a fixed cascade of nodes.

7.6 Protocol enhancements

We describe two optional extensions of LightMix that extend its functionality and improve its efficiency.

7.6.1 Return path

It is easy to extend LightMix to enable a receiver to send an immediate response through the mix-net, for example, to acknowledge receiving a message. To accomplish this goal, the nodes generate additional random values $\mathbf{s}'$ and compute the permuted products $\mathbf{S}'$ during the precomputation phase. Also, the nodes and users generate fresh keys $\mathbf{k}'$, that will be used to encrypt the return message.

For a return path, the mixnodes apply the inverse permutations π^{-1} so that the responses will arrive at the original senders. Unless the recipient who sends a

response shares keys with the system, no fresh $\mathbf{r}'$ values are needed because the message would not enter the system blinded and hence Step 1 of the real-time phase could be skipped. In Step 2, the system applies the inverse permutations π^{-1} and fresh $\mathbf{s}'$ values. In Step 3, to encrypt the response to the original sender, instead of multiplying only with its decryption component, each node multiplies with the product of its decryption component and $\mathbf{k}'$ value.

7.6.2 Network handler

Introducing an untrusted *network handler* reduces latency. In Steps 1 and 3 of the precomputation and real-time phases, only products of values known by the individual nodes are computed (see Section 7.3.2). To compute these products, it is not necessary to make a full pass through the mix-net. Instead, each node can send its values to an untrusted third party, which we will call the network handler, who can compute the products and return the results to the mix-net. Doing so reduces latency of the network significantly because each node can send its values simultaneously to the handler, instead of forwarding its local result to the next node sequentially. The network handler does not learn any secret value, and it computes only values that would anyway become public.

The network handler, and each of the mixnodes, is a single point of failure. In the event of failure, however, because the handler performs only public operations, it can be easily replaced by another entity—for example, by one of the mixnodes.

7.7 Comparison with other mix-nets

We compare LightMix with well-known fixed-cascade mix-net approaches based on performance. Specifically, as summarized in Table 7.1, we compare the performance of the core LightMix protocol with that of each of the following three competing approaches: original mix-net and hybrid mix-nets, re-encryption mix-net, and re-encryption mix-net with precomputation.

For each approach, we compare the precomputation and real-time costs, further compared by number of single-party public-key operations, multi-party public-key operations, and multiplications (each by client and by mixnodes). Note that, when using ElGamal encryption, the multiplication of two ciphertexts requires two multiplications.

The original mix-net [80] requires each sender to perform n encryptions; each of the n mixnodes in the cascade performs one decryption per message and β decryptions per batch. In total, all mixnodes perform n decryptions per message and βn decryptions per batch. Hybrid mix-nets [242, 183] require the same amount of asymmetric encryptions, but on a smaller plaintext.

In re-encryption mix-nets [250], each client performs one encryption of its message using the mix-net's shared public key. Each node re-randomizes every message, instead of decrypting each one as with original mix-nets, resulting in one public-key operation and one multiplication of ciphertexts per message per node. In addition, the nodes need to decrypt the output of the mix-net in a multi-party computation.

Re-encryption mix-nets can be improved in a straightforward way using precomputation to perform the public-key operations necessary for the re-randomization, similarly to LightMix's strategy. As shown in Table 7.1, however, with regard to real-time computation, LightMix outperforms re-encryption mix-nets with precomputation.

Universal re-encryption mix-nets perform much more slowly because they require senders to encrypt messages with public keys of their recipients.

Table 7.1: Performance comparison of the core LightMix protocol with three competing mix-net approaches: number of operations performed for a batch of β messages processed by an n-node mix-net. One multi-party public-key (PK) operation requires all nodes to participate.

	Precomputation (operations for all mixnodes)			Real time (operations per client; operations for all mixnodes)		
	PK operations	Multi-party PK operations	Multiplications	PK operations	Multi-party PK operations	Multiplications
LightMix (core)	$2n\beta$	β	$4n - 2\beta$	0; 0	0; 0	n; $3n\beta$
Original mix	-	-	-	n; $n\beta$	0; 0	0; 0
Re-encryption mix	-	-	-	1; $n\beta$	0; β	0; $2n\beta$
Re-encryption mix (precomputation)	$n\beta$	0	0	1; 0	0; β	0; $2n\beta$

7.8 Proof of concept

We implemented a proof-of-concept protoype in Python, including tagging detection and the network handler, as discussed in Sections 7.4.2 and 7.6.2, respectively. Each mixnode includes a keyserver (to establish shared keys with the users) and a mix-net server (to carry out the precomputations and real-time computations). For the precomputation, each mixnode uses parallel processes for the computation of the encryptions and the decryption shares. In the real-time phase, all operations are performed in a single thread.

We performed experiments by running the protoype on Amazon Web Services (AWS) instances, with each node comprising a c3.large with two virtual processors and 3.75 GB of RAM. For all values, we used a prime-order group of 2048 bits.

On the AWS instances, each 2048-bit ElGamal encryption took approximately 10 milliseconds on average, and the computation of a decryption share took approximately 5 milliseconds. Multiplication of group elements took only a fraction of a millisecond.

For our experiments we performed 100 precomputation and real-time phases for selected batch sizes up to 1000 with five mixnodes. Table 7.2 gives observed timings on the network handler for selected batch sizes using five mixnodes, without any enhanced security mechanisms. We measured elapsed time on the network handler from the time it instructs the nodes to start until either the precomputation finished successfully, or until it computed the final responses to be sent to the users in the real-time phase. Table 7.3 gives timings for the real-time phase per node and for the network handler, in both CPU and wall clock time. These timings show the low computational load on the nodes during this phase.

These timings demonstrate the high performance of the system in the real-time phase. The precomputation can be easily accelerated by performing more computations in parallel. Additional processors would significantly improve the time it takes to compute all necessary encryptions and decryption shares. For the real-time phase, a network connection with low latency would improve the timings.

7.9 Discussion and future work

We now briefly discuss how to arrange messages into batches and how to deal with node failures. We also outline some of our future plans.

Table 7.2: Timings measured on the network handler from the start of the phases until the final values or responses are computed. Timings are in seconds (wall clock) for 100 runs of the precomputation and real-time phases, for various batch sizes using five mixnodes.

Batch	Precomputation		Real time	
size	Mean	Std. dev.	Mean	Std. dev.
10	0.48	0.07	0.07	0.02
50	1.99	0.04	0.21	0.04
100	4.00	0.30	0.38	0.06
200	7.74	0.09	0.75	0.08
300	11.46	0.13	1.09	0.08
400	15.24	0.11	1.44	0.08
500	19.08	0.23	1.80	0.11
1000	37.94	0.19	3.58	0.12

Table 7.3: Mean timings in seconds (CPU and wall clock) for 100 runs of the real-time phase of the LightMix protocol measured on the mixnodes and network handler, for various batch sizes using five mixnodes. For the mixnodes the mean time is taken over all mixnodes.

Batch	Mixnode		Network handler	
size	CPU	Wall	CPU	Wall
10	0.01	0.04	0.01	0.07
50	0.04	0.16	0.02	0.21
100	0.07	0.30	0.02	0.38
200	0.14	0.60	0.04	0.75
300	0.22	0.87	0.06	1.09
400	0.29	1.15	0.07	1.44
500	0.36	1.45	0.09	1.80
1000	0.73	2.88	0.19	3.58

Batch strategy. LightMix follows the "threshold and timed mixing strategy" from Serjantov et al. [286], where a new round is started every t seconds only if there are at least β messages in the buffer. We expect at least β users to be active at any given time. When a smaller number of users is active, this strategy can lead to increased latency or even disruption. At the cost of increased energy consumption, one design choice is to inject dummy messages when needed to ensure enough

traffic to have β messages every t seconds. Alternatively, empty slots can be used to verify if the precomputation was performed correctly, by revealing all committed values for these slots, where empty slots to use should be chosen at random.

Node failure. Because LightMix uses a fixed cascade of nodes, it is important to consider what happens if a node fails. First, we consider a node failure to be a highly rare event because we expect each node to be a highly reliable computing service that is capable of seamlessly handling failures. Second, the system will detect node failure and notify the senders and the other nodes; senders will be instructed to resend using a new cascade (e.g., the old cascade without the failed node). Each node can detect failures by listening for periodic "pings" from the other nodes.

To minimize possible disruption caused by a single failure, at the cost of increasing the precomputations, the following option can be deployed: Each node can have a reserve of precomputations ready to use for certain alternative cascades. For example, this reserve can include each of the alternative cascades formed by removing any one node from the current cascade.

7.10 Conclusion

LightMix builds on the strong anonimity properties of mix-nets, and improves real-time cryptographic performance by eliminating real-time public-key operations in its core protocol. By replacing real-time public-key operations with precomputations, and by avoiding the user's direct involvement with the construction of the path through the mixnodes, LightMix reduces computational costs for users. Even though the adversary may know all senders and receivers in each batch, she cannot link any sender and receiver unless all mixnodes are compromised. LightMix enables lightweight devices and smartphones to communicate anonymously using less battery power and with higher real-time performance.

Bibliography

[1] Ben Adida and Douglas Wikström. Offline/Online Mixing. In *ICALP 2007*, pages 484–495, 2007. (Cited on page 24)

[2] Charu C. Aggarwal, Naveen Ashish, and Amit Sheth. *The Internet of Things: A Survey from the Data-Centric Perspective*, pages 383–428. Springer US, 2013. (Cited on page 34)

[3] Philip E. Agre and Marc Rotenberg. *Technology and Privacy: the New Landscape*. The MIT Press, 1997. (Cited on page 14)

[4] Alcaide Almudena, Esther Palomar, José Montero-Castillo, and Arturo Ribagorda. Anonymous Authentication for Privacy-preserving IoT Target-driven Applications. *Computers and Security*, 37:111–123, 2013. (Cited on page 34)

[5] Gergely Alpár. *Attribute-Based Identity Management*. PhD thesis, Digital Security group, Radboud University, 2015. (Cited on page 38)

[6] Gergely Alpár, Lejla Batina, Lynn Batten, Veelasha Moonsamy, Anna Krasnova, Antoine Guellier, and Iynkaran Natgunanathan. new directions in IoT privacy using attribute-based authentication. In *Proc. ACM International Conference on Computing Frontiers*. (Cited on page 136)

[7] Gergely Alpár, Lejla Batina, Lynn Batten, Veelasha Moonsamy, Anna Krasnova, Antoine Guellier, and Iynkaran Natgunanathan. New Directions in IoT Privacy Using Attribute-based Authentication. In *Proceedings of the ACM International Conference on Computing Frontiers*, CF '16, pages 461–466, New York, NY, USA, 2016. ACM. (Cited on page 9)

[8] Gergely Alpár, Lejla Batina, and Wouter Lueks. Designated Attribute-Based Proofs for RFID Applications. In *Proc. Workshop on RFID Security and Privacy*

(RFIDSec 2013), volume 7739 of *Selected Notes in Computer Science*, pages 59–75, Nijmegen, The Netherlands, 2012. Springer Berlin Heidelberg. (Cited on pages 35 and 68)

[9] Gergely Alpár and Jaap-Henk Hoepman. A Secure Channel for Attribute-based Credentials. In *Proc. Eighth ACM workshop on digital identity management (DIM 2013)*, pages 13–18, Berlin, Germany, 2013. ACM. (Cited on page 35)

[10] Gergely Alpár, Jaap-Henk Hoepman, and Johanneke Siljee. The Identity Crisis – Security, Privacy and Usability Issues in Identity Management. *Journal of Information System Security*, 9(1):23–53, 2013. (Cited on page 34)

[11] Gergely Alpár and Bart Jacobs. Credential Design in Attribute-Based Identity Management. In *Proc. Bridging distances in technology and regulation, 3rd TILTing Perspectives Conference, 2013*, pages 189–204. Wolf Legal Publishers, 2013. (Cited on pages 33 and 40)

[12] Scott Amyx, Arun Aryasomajula, Brenda Bannan, David Bartlett, et al. *Internet of Things and Data Analytics Handbook*. "John Wiley & Sons", Hoboken, New Jersey, 2017. (Cited on page 30)

[13] Jacob Appelbaum and Roger Dingledine. How Governments have Tried to Block Tor. website, 2011 accessed 2017. http://ftp.ccc.de/congress/28C3/mp4-h264-HQ/28c3-4800-en-how_governments_have_tried_to_block_tor_h264.mp4. (Cited on pages 96 and 109)

[14] Apple. iOS Security. website, 2012 accessed 2017. http://images.apple.com/iphone/business/docs/iOS_Security_Oct12.pdf. (Cited on page 127)

[15] Alex Arbit, Yoel Livne, Yossef Oren, and Avishai Wool. Implementing Public-key Cryptography on Passive RFID Tags is Practical. *International Journal of Information Security*, 14(1):85–99, 2015. (Cited on page 136)

[16] Apple's iPhone Already Has a Backdoor. website, 2016 (accessed January 2017). https://apple.slashdot.org/story/16/02/22/1518202/apples-iphone-already-has-a-backdoor/. (Cited on page 4)

[17] Gilad Asharov, Yehuda Lindell, Thomas Schneider, and Michael Zohner. More Efficient Oblivious Transfer and Extensions for Faster Secure Computation. In *Proc. 20th ACM Conference on Computer and Communications Security (CCS 2013)*, pages 535–548, New York, NY, USA, 2013. ACM. (Cited on page 19)

[18] Kevin Ashton. That 'Internet of Things' Thing. website, 2009 (accessed January 2017). http://www.rfidjournal.com/articles/view?4986. (Cited on page 29)

[19] Jean-Philippe Aumasson, Luca Henzen, Willi Meier, and María Naya-Plasencia. Quark: A Lightweight Hash. *Journal of Cryptology*, 26(2):313–339, 2013. (Cited on page 50)

[20] Gildas Avoine. Adversarial Model for Radio Frequency Identification. Cryptology ePrint Archive, Report 2005/049, 2005. http://eprint.iacr.org/2005/049. (Cited on page 53)

[21] Gildas Avoine, Iwen Coisel, and Tania Martin. Untraceability Model for RFID. *IEEE Transactions on Mobile Computing*, 13(10):2397–2405, October 2014. (Cited on page 53)

[22] M. Backes, I. Goldberg, A. Kate, and E. Mohammadi. Provably Secure and Practical Onion Routing. In *Proc. 25th IEEE Computer Security Foundations Symposium (CSF)*, 2012. (Cited on page 24)

[23] Michael Backes, Aniket Kate, and Esfandiar Mohammadi. Ace: An Efficient Key-exchange Protocol for Onion Routing. In *Proc. 11th ACM Workshop on Privacy in the Electronic Society (WPES)*, pages 55–64, 2012. (Cited on page 24)

[24] Rajiv Bagai, Bin Tang, Ahsan A. Khan, and Abdus Samad. A System-wide Anonymity Metric with Message Multiplicities. *Int. J. Secur. Netw.*, 10(1):20–31, April 2015. (Cited on page 15)

[25] Lejla Batina, Jens Hermans, Jaap-Henk Hoepman, and Anna Krasnova. *High-Speed Dating Privacy-Preserving Attribute Matching for RFID*, pages 19–35. Springer International Publishing, Oxford, UK, 2014. (Cited on page 10)

[26] Japanese Police Target Users of Tor Anonymous Network. website, 2013 accessed 2017. http://www.bbc.com/news/technology-22248692. (Cited on page 95)

[27] A. Beimel and Y. Stahl. Robust Information-theoretic Private Information Retrieval. In S. Cimato C. Galdi G. Persiano, editor, *Proc. 3rd Conference on Security in Communication Networks*, volume 2576 of *Selected Notes in Computer Science*, pages 326–341. Springer-Verlag Berlin Heidelberg, 2002. (Cited on page 19)

[28] Amos Beimel and Yuval Ishai. Information-Theoretic Private Information Retrieval: A Unified Construction. *Selected Notes in Computer Science*, 2076:89–98, 2001. (Cited on page 19)

[29] Amos Beimel, Yuval Ishai, and Tal Malkin. Reducing the Servers Computation in Private Information Retrieval: PIR with Preprocessing. In *Proc. Advances in Cryptology (CRYPTO 2000)*, volume 1880 of *Selected Notes in Computer Science*, pages 55–73. Springer-Verlag Berlin Heidelberg, 2000. (Cited on page 19)

[30] Belkin WeMo. website, accessed January 2017. http://www.belkin.com/us/Products/home-automation/c/wemo-home-automation/. (Cited on pages 4 and 30)

[31] Mihir Bellare, Alexandra Boldyreva, and Adriana Palacio. An Uninstantiable Random-Oracle-Model Scheme for a Hybrid-Encryption Problem. In *Proc. Advances in Cryptology (EUROCRYPT 2004)*, volume 3027 of *Selected Notes in Computer Science*, pages 171–188. Springer Berlin Heidelberg, 2004. (Cited on page 50)

[32] Mihir Bellare, Anand Desai, E. Jokipii, and Phillip Rogaway. A Concrete Security Treatment of Symmetric Encryption. In *Proc. 38th Annual Symposium on Foundations of Computer Science (FOCS 1997)*, pages 394–403. IEEE Press., 1997. (Cited on page 110)

[33] Mihir Bellare, Anand Desai, David Pointcheval, and Phillip Rogaway. Relations Among Notions of Security for Public-Key Encryption Schemes. In *Proc. Advances in Cryptology (CRYPTO 1998)*, volume 1462 of *Selected Notes in Computer Science*, pages 26–45. Springer Berlin Heidelberg, 1998. (Cited on page 47)

[34] Mihir Bellare, Juan A Garay, Ralf Hauser, Amir Herzberg, Hugo Krawczyk, Michael Steiner, Gene Tsudik, Els Van Herreweghen, and Michael Waidner. Design, Implementation, and Deployment of the iKP Secure Electronic Payment System. *IEEE Journal on selected areas in communications*, 18(4):611–627, 2000. (Cited on page 18)

[35] Mihir Bellare, Oded Goldreich, and Anton Mityagin. The Power of Verification Queries in Message Authentication and Authenticated Encryption. Cryptology ePrint Archive, Report 2004/309, 2004. http://eprint.iacr.org/2004/309/. (Cited on pages 51 and 59)

[36] Mihir Bellare, Joe Kilian, and Phillip Rogaway. The Security of the Cipher Block Chaining Message Authentication Code. *Journal of Computer and System Sciences*, 61(3):362–399, 2000. (Cited on pages 51 and 59)

[37] Mihir Bellare and Phillip Rogaway. Random Oracles Are Practical: A Paradigm for Designing Efficient Protocols. In *Proc. 1st ACM Conference on Computer and Communications Security (CCS 1993)*, pages 62–73. ACM, 1993. (Cited on page 49)

[38] Eli Ben-Sasson, Alessandro Chiesa, Christina Garman, Matthew Green, Ian Miers, Eran Tromer, and Madars Virza. Zerocash: Practical Decentralized Anonymous E-Cash from Bitcoin. In *Proc. 35th IEEE Symposium on Security and Privacy (SP 2014)*. IEEE Press., May 2014. (Cited on page 19)

[39] Josh Benaloh. Simple Verifiable Elections. In *Proc. USENIX Accurate Electronic Voting Technology Workshop (EVT)*, pages 5–5, 2006. (Cited on page 140)

[40] Mike Bendel. Hackers Describe PS3 Security As Epic Fail, Gain Unrestricted Access. website, 2010 accessed 2017. https://www.exophase.com/20540/hackers-describe-ps3-security-as-epic-fail-gain-unrestricted-access/. (Cited on page 50)

[41] Krista Bennett and Christian Grothoff. GAP – Practical Anonymous Networking. In Roger Dingledine, editor, *Proc. Privacy Enhancing Technologies Workshop (PET 2003)*, pages 141–160. Springer-Verlag Berlin Heidelberg, March 2003. (Cited on page 19)

[42] Daniel J. Bernstein. Current consensus on ECC, 2001 accessed 2017. https://groups.google.com/forum/message/raw?msg=sci.crypt/mu_paShEU3w/m491pYxHbtAJ. (Cited on page 126)

[43] Daniel J. Bernstein. Curve25519: New Diffie-Hellman Speed Records. In *Proc. Public Key Cryptography (PKC 2006)*, volume 3958 of *Selected Notes in Computer Science*, pages 207–228. Springer Berlin Heidelberg, 2006. (Cited on pages 102, 123, 125, and 127)

[44] Daniel J. Bernstein. Multi-user Schnorr Security, Revisited. Cryptology ePrint Archive, Report 2015/996, 2015. http://eprint.iacr.org/2015/996. (Cited on page 112)

[45] Daniel J. Bernstein, Peter Birkner, Marc Joye, Tanja Lange, and Christiane Peters. Twisted Edwards Curves. In *Proc. Progress in Cryptology (AFRICACRYPT 2008)*, volume 5023 of *Selected Notes in Computer Science*, pages 389–405. Springer Berlin Heidelberg, 2008. (Cited on pages 101, 125, and 127)

[46] Daniel J. Bernstein, Peter Birkner, Tanja Lange, and Christiane Peters. Optimizing Double-Base Elliptic-Curve Single-Scalar Multiplication. In *Proc. Progress in Cryptology (INDOCRYPT 2007)*, volume 4859 of *Selected Notes in Computer Science*, pages 167–182. Springer Berlin Heidelberg, 2007. (Cited on page 99)

[47] Daniel J. Bernstein, Niels Duif, Tanja Lange, Peter Schwabe, and Bo-Yin Yang. High-speed High-security Signatures. In *Proc. Cryptographic Hardware and Embedded Systems (CHES 2011)*, volume 6917 of *Selected Notes in Computer Science*, pages 124–142. Springer Berlin Heidelberg, 2011. (Cited on pages 112, 127, and 132)

[48] Daniel J. Bernstein, Mike Hamburg, Anna Krasnova, and Tanja Lange. Elligator: Elliptic-curve Points Indistinguishable from Uniform Random Strings. In *Proceedings of the 2013 ACM SIGSAC conference on Computer & Communications Security*, CCS '13, pages 967–980, New York, NY, USA, 2013. ACM. (Cited on page 11)

[49] Daniel J. Bernstein and Tanja Lange. Faster Addition and Doubling on Elliptic Curves. In *Proc. Advances in Cryptology (ASIACRYPT 2007)*, volume 4833 of *Selected Notes in Computer Science*, pages 29–50. Springer Berlin Heidelberg, 2007. (Cited on pages 99, 100, 124, 125, and 127)

[50] Daniel J. Bernstein and Peter Schwabe. NEON Crypto. In *Proc. Cryptographic Hardware and Embedded Systems (CHES 2012)*, volume 7428 of *Selected Notes in Computer Science*, pages 320–339. Springer Berlin Heidelberg, 2012. (Cited on page 127)

[51] Oliver Berthold and Heinrich Langos. Dummy Traffic against Long Term Intersection Attacks. In *Proc. Privacy Enhancing Technologies Workshop (PET 2002)*, volume 2482 of *Selected Notes in Computer Science*, pages 110–128. Springer-Verlag Berlin Heidelberg, 2002. (Cited on page 155)

[52] Oliver Berthold, Andreas Pfitzmann, and Ronny Standtke. The Disadvantages of Free MIX Routes and how to Overcome Them. In *Proc. International Work-*

shop on Design Issues in Anonymity and Unobservability, volume 2009 of *Selected Notes in Computer Science*, pages 30–45. Springer-Verlag Berlin Heidelberg, 2000. (Cited on pages 135, 136, 154, and 155)

[53] Ingrid Biehl, Bernd Meyer, and Volker Müller. Differential Fault Attacks on Elliptic Curve Cryptosystems. In *Proc. Advances in Cryptology (CRYPTO 2000)*, volume 1880 of *Selected Notes in Computer Science*, pages 131–146. Springer Berlin Heidelberg, 2000. (Cited on page 127)

[54] Protect your privacy. website, accessed August 2017. https://bitcoin.org/en/protect-your-privacy. (Cited on page 19)

[55] Cecylia Bocovich and Ian Goldberg. Slitheen: Perfectly Imitated Decoy Routing Through Traffic Replacement. In *Proc. 23rd ACM Conference on Computer and Communications Security (CCS 2016)*, pages 1702–1714, New York, NY, USA, 2016. ACM. (Cited on page 20)

[56] Dan Boneh and Matthew K. Franklin. Identity-Based Encryption from the Weil Pairing. In *Proc. Advances in Cryptology (CRYPTO 2001)*, volume 2139 of *Selected Notes in Computer Science*, pages 213–229. Springer Berlin Heidelberg, 2001. (Cited on page 106)

[57] Joseph Bonneau, Arvind Narayanan, Andrew Miller, Jeremy Clark, Joshua A Kroll, and Edward W Felten. Mixcoin: Anonymity for Bitcoin with Accountable Mixes. In *Proc. International Conference on Financial Cryptography and Data Security*, pages 486–504. Springer-Verlag Berlin Heidelberg, 2014. (Cited on page 19)

[58] Jurjen N. Bos and Bert den Boer. Detection of Disrupters in the DC Protocol. In *Proc. Advances in Cryptology (EUROCRYPT 1989)*, volume 434 of *Selected Notes in Computer Science*, pages 320–327. Springer Berlin Heidelberg, 1989. (Cited on pages 74 and 91)

[59] Colin Boyd, Paul Montague, and Khanh Quoc Nguyen. Elliptic Curve Based Password Authenticated Key Exchange Protocols. In *Information Security and Privacy: 6th Australasian Conference (ACISP 2001)*, volume 2119 of *Selected Notes in Computer Science*, pages 487–501. Springer Berlin Heidelberg, 2001. (Cited on pages 106 and 114)

[60] Brainpool. ECC Brainpool Standard Curves and Curve Generation, v. 1.0, 2005 accessed 2017. http://www.ecc-brainpool.org/download/Domain-parameters.pdf. (Cited on pages 98 and 125)

[61] Stefan A. Brands. *Rethinking Public Key Infrastructures and Digital Certificates: Building in Privacy*. MIT Press, Cambridge, MA, USA, 2000. (Cited on page 18)

[62] Gilles Brassard, Claude Crépeau, and Jean-Marc Robert. All-or-Nothing Disclosure of Secrets. In *Proc. Advances in Cryptology (CRYPTO 1986)*, volume 263 of *Selected Notes in Computer Science*, pages 234–238. Springer-Verlag Berlin Heidelberg, 1986. (Cited on page 19)

[63] Michael Braun, Erwin Hess, and Bernd Meyer. Using Elliptic Curves on RFID Tags. *IJCSNS International Journal of Computer Science and Network Security*, 8(2):1–9, 2008. (Cited on page 68)

[64] Justin Brickell and Vitaly Shmatikov. Efficient Anonymity-preserving Data Collection. In *Proc. Twelfth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining 2006*, pages 76–85, 2006. (Cited on page 152)

[65] Eric Brier, Jean-Sébastien Coron, Thomas Icart, David Madore, Hugues Randriam, and Mehdi Tibouchi. Efficient Indifferentiable Hashing into Ordinary Elliptic Curves. In *Proc. Advances in Cryptology (CRYPTO 2010)*, volume 6223 of *Selected Notes in Computer Science*, pages 237–254. Springer Berlin Heidelberg, 2010. (Cited on page 106)

[66] Eric Brier and Marc Joye. Weierstraß Elliptic Curves and Side-Channel Attacks. In *Proc. Public Key Cryptography (PKC 2003)*, volume 2274 of *Selected Notes in Computer Science*, pages 335–345. Springer Berlin Heidelberg, 2002. (Cited on page 103)

[67] Julien Bringer, Hervé Chabanne, and Thomas Icart. Cryptanalysis of EC-RAC, a RFID Identification Protocol. In *Proc. Cryptology and Network Security: 7th International Conference (CANS 2008)*, volume 5339 of *Selected Notes in Computer Science*, pages 149–161. Springer-Verlag Berlin Heidelberg, 2008. (Cited on page 68)

[68] Jon Brodkin. Iran Reportedly Blocking Encrypted Internet Traffic. website, 2012 accessed 2017. http://arstechnica.com/tech-policy/2012/02/iran-reportedly-blocking-encrypted-internet-traffic/. (Cited on page 96)

[69] Johannes Buchmann, Erik Dahmen, and Andreas Hülsing. XMSS – A Practical Forward Secure Signature Scheme Based on Minimal Security Assumptions. In *Post-Quantum Cryptography: 4th International Workshop (PQCrypto 2011)*, volume 7071 of *Selected Notes in Computer Science*, pages 117–129. Springer-Verlag Berlin Heidelberg, 2011. (Cited on page 50)

[70] Chase Buckle. Digital Consumers Own 3.64 Connected Devices. website, 2016 (accessed January 2017). https://www.globalwebindex.net/blog/digital-consumers-own-3.64-connected-devices. (Cited on page 30)

[71] Christian Cachin, Silvio Micali, and Markus Stadler. Computationally Private Information Retrieval with Polylogarithmic Communication. In Jacques Stern, editor, *Proc. Advances in Cryptology: Workshop on the Theory and Application of Cryptographic Techniques (EUROCRYPT 1999)*, volume 1592 of *Selected Notes in Computer Science*, pages 402–414. Springer-Verlag Berlin Heidelberg, 1999. (Cited on page 19)

[72] J. Camenisch, A. Lysyanskaya, and M. Meyerovich. Endorsed e-cash. In *Proc. 28th IEEE Symposium on Security and Privacy (SP 2007)*, pages 101–115, May 2007. (Cited on page 18)

[73] Jan Camenisch and Anna Lysyanskaya. An Efficient System for Non-transferable Anonymous Credentials with Optional Anonymity Revocation. In *Proc. Advances in Cryptology (EUROCRYPT 2001)*, pages 93–118. Springer-Verlag Berlin Heidelberg, 2001. (Cited on page 18)

[74] Jan Camenisch and Anna Lysyanskaya. A Formal Treatment of Onion Routing. In *Proc. Advances in Cryptology (CRYPTO 2005)*, volume 3621 of *Selected Notes in Computer Science*, pages 169–187. Springer Berlin Heidelberg, 2005. (Cited on page 24)

[75] Jan Camenisch and Els Van Herreweghen. Design and Implementation of the Idemix Anonymous Credential System. In *Proc. 9th ACM Conference on Computer and Communications Security (CCS 2002)*, pages 21–30. ACM, 2002. (Cited on pages 35 and 37)

[76] Ran Canetti, Oded Goldreich, and Shai Halevi. The Random Oracle Methodology, Revisited. *Journal of the ACM (JACM)*, 51(4):557–594, 2004. (Cited on page 50)

[77] Jianneng Cao and Panagiotis Karras. Publishing Microdata with a Robust Privacy Guarantee. *Proc. VLDB Endow.*, 5(11):1388–1399, July 2012. (Cited on page 21)

[78] John I. Capetanakis. Tree Algorithms for Packet Broadcast Channels. *IEEE Transactions on Information Theory*, IT-25(5):505–515, 1979. (Cited on page 74)

[79] David W. Chadwick. Federated Identity Management. In *Foundations of Security Analysis and Design V (FOSAD 2007/2008/2009)*, volume 5705 of *Selected Notes in Computer Science*, pages 96–120. Springer Berlin Heidelberg, 2009. (Cited on pages 34 and 35)

[80] David Chaum. Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms. *Communications of the ACM*, 24(2):84–90, 1981. (Cited on pages 18, 20, 21, 22, 28, 41, 71, 145, 154, and 157)

[81] David Chaum. Blind Signatures for Untraceable Payments. In *Proc. Advances in Cryptology (CRYPTO 1983)*, pages 199–203. Springer US, 1983. (Cited on page 18)

[82] David Chaum. Security Without Identification: Transaction Systems to Make Big Brother Obsolete. *Communications of the ACM*, 28(10):1030–1044, October 1985. (Cited on page 18)

[83] David Chaum. The Dining Cryptographers Problem: Unconditional Sender and Recipient Untraceability. *Journal of Cryptology*, 1(1):65–75, 1988. (Cited on pages 18, 21, 25, 26, 72, 73, 78, 79, and 91)

[84] David Chaum, Amos Fiat, and Moni Naor. *Untraceable Electronic Cash*, pages 319–327. Springer New York, New York, NY, 1988. (Cited on page 18)

[85] Chen Chen, Daniele Enrico Asoni, David Barrera, George Danezis, and Adrian Perrig. HORNET: High-speed Onion Routing at the Network Layer. In *Proc. 22nd ACM Conference on Computer and Communications Security (CCS 2015)*, pages 1441–1454. ACM, 2015. (Cited on page 24)

[86] Benny Chor, Eyal Kushilevitz, Oded Goldreich, and Madhu Sudan. Private Information Retrieval. *Journal of the ACM (JACM)*, 45(6):965–981, November 1998. (Cited on pages 19 and 27)

[87] Ian Clarke, Oskar Sandberg, Brandon Wiley, and Theodore W. Hong. Freenet: A Distributed Anonymous Information Storage and Retrieval System. In

Designing Privacy Enhancing Technologies, volume 2009 of *Selected Notes in Computer Science*, pages 46–66. Springer-Verlag Berlin Heidelberg, July 2001. (Cited on page 19)

[88] Chris Clifton, Murat Kantarcioglu, Jaideep Vaidya, Xiaodong Lin, and Michael Y. Zhu. Tools for Privacy Preserving Distributed Data Mining. *SIGKDD Explor. Newsl.*, 4(2):28–34, December 2002. (Cited on page 21)

[89] Chris Clifton and Tamir Tassa. On Syntactic Anonymity and Differential Privacy. *Trans. Data Privacy*, 6(2):161–183, August 2013. (Cited on page 21)

[90] Google Cloud Platform. web-site, accessed January 2017. https://cloud.google.com/solutions/iot/. (Cited on page 31)

[91] Oracle Cloud. web-site, accessed January 2017. https://cloud.oracle.com/iot. (Cited on page 31)

[92] Henri Cohen, Gerhard Frey, Roberto Avanzi, Christophe Doche, Tanja Lange, Kim Nguyen, and Frederik Vercauteren. *Handbook of Elliptic and Hyperelliptic Curve Cryptography, Second Edition*. Chapman & Hall/CRC, 2nd edition, 2012. (Cited on page 100)

[93] Henry Corrigan-Gibbs and Bryan Ford. Dissent: Accountable Anonymous Group Messaging. In *Proc. 17th ACM Conference on Computer and Communications Security (CCS 2010)*, pages 340–350. ACM, 2010. (Cited on pages 25, 75, and 91)

[94] Henry Corrigan-Gibbs, David I. Wolinsky, and Bryan Ford. Proactively Accountable Anonymous Messaging in Verdict. In *Proc. 22nd USENIX Security Symposium (USENIX 2013)*, pages 147–162. USENIX Association, 2013. (Cited on pages 25, 75, and 91)

[95] Araujo Laécio Costa and Ruy J. B. de Queiroz. The Use of Fully Homomorphic Encryption in Data Mining with Privacy Preserving. In *Proc. 2014 International Conference on Advances in Big Data Analytics*, volume I, 2014. (Cited on page 21)

[96] Craig Costello, Patrick Longa, and Michael Naehrig. A Brief Discussion on Selecting New Elliptic Curves. Technical Report MSR-TR-2015-46, Microsoft, June 2015. Position paper presented at the NIST Workshop on Elliptic Curve Cryptography Standards (http://www.nist.gov/itl/csd/ct/ecc-workshop.cfm). (Cited on page 98)

[97] Neil Costigan and Peter Schwabe. Fast Elliptic-Curve Cryptography on the Cell Broadband Engine. In *Proc. Progress in Cryptology (AFRICACRYPT 2009)*, volume 5580 of *Selected Notes in Computer Science*, pages 368–385. Springer Berlin Heidelberg, 2009. (Cited on page 127)

[98] Ronald Cramer and Victor Shoup. Design and Analysis of Practical Public-Key Encryption Schemes Secure Against Adaptive Chosen Ciphertext Attack. *SIAM Journal on Computing (SICOMP)*, 33(1):167–226, 2004. (Cited on page 111)

[99] George Danezis. The Traffic Analysis of Continuous-Time Mixes. In *Proc. Privacy Enhancing Technologies*, volume 3424 of *Selected Notes in Computer Science*, pages 35–50. Springer-Verlag Berlin Heidelberg, 2004. (Cited on pages 135 and 155)

[100] George Danezis and Claudia Diaz. A Survey of Anonymous Communication Channels. Technical report, Microsoft, February 2008. (Cited on page 21)

[101] George Danezis, Claudia Díaz, and Carmela Troncoso. Two-Sided Statistical Disclosure Attack. In *Proc. Privacy Enhancing Technologies Symposium (PETS 2007)*, volume 4776 of *Selected Notes in Computer Science*, pages 30–44. Springer-Verlag Berlin Heidelberg, 2007. (Cited on pages 135 and 155)

[102] George Danezis, Cedric Fournet, Markulf Kohlweiss, and Bryan Parno. Pinocchio Coin: Building Zerocoin from a Succinct Pairing-based Proof System. In *Proc. First ACM workshop on Language support for privacy-enhancing technologies*, pages 27–30. ACM, 2013. (Cited on page 19)

[103] George Danezis and Ian Goldberg. Sphinx: A Compact and Provably Secure Mix Format. In *Proc. 30th IEEE Symposium on Security and Privacy (SP 2009)*, pages 269–282. IEEE Press., 2009. (Cited on page 41)

[104] George Danezis and Andrei Serjantov. Statistical Disclosure or Intersection Attacks on Anonymity Systems. In *Proc. Information Hiding (IH 2004)*, volume 3200 of *Selected Notes in Computer Science*, pages 293–308. Springer-Verlag Berlin Heidelberg, 2004. (Cited on pages 135 and 155)

[105] Scott Vanstone Darrel Hankerson, Alfred J. Menezes. *Guide to Elliptic Curve Cryptography*. Springer-Verlag New York, 1 edition, 2004. (Cited on page 100)

[106] Ton Deursen and Saša Radomirović. Insider Attacks and Privacy of RFID Protocols. In *Proc. Public Key Infrastructures, Services and Applications: 8th European*

Workshop (EuroPKI 2011), volume 7163 of *Selected Notes in Computer Science*, pages 91–105. Springer-Verlag Berlin Heidelberg, 2012. (Cited on page 68)

[107] Casey Devet and Ian Goldberg. The Best of Both Worlds: Combining Information-Theoretic and Computational PIR for Communication Efficiency. In *Proc. Privacy Enhancing Technologies Symposium (PETS 2014)*, July 2014. (Cited on page 19)

[108] Giovanni Di Crescenzo, Tal Malkin, and Rafail Ostrovsky. Single Database Private Information Retrieval Implies Oblivious Transfer. In *Proc. Advances in Cryptology (EUROCRYPT 2000)*, volume 1807 of *Selected Notes in Computer Science*, pages 122–138. Springer-Verlag Berlin Heidelberg, 2000. (Cited on page 19)

[109] Claudia Díaz, Stefaan Seys, Joris Claessens, and Bart Preneel. Towards Measuring Anonymity. In *Proc. Privacy Enhancing Technologies Workshop (PET 2002)*, volume 2482 of *Selected Notes in Computer Science*, pages 54–68. Springer-Verlag Berlin Heidelberg, 2002. (Cited on page 15)

[110] Claus Diem. The GHS Attack in Odd Characteristic. *J. Ramanujan Mathematical Society*, 18:1–32, 2003. (Cited on page 125)

[111] R. Dingledine and N. Mathewson. Tor Protocol Specification. `https://gitweb.torproject.org/torspec.git?a=blob_plain;hb=HEAD;f=tor-spec.txt`, 2008. Accessed Nov 2011. (Cited on page 135)

[112] Roger Dingledine, Michael J. Freedman, and David Molnar. The Free Haven Project: Distributed Anonymous Storage Service. In H. Federrath, editor, *Proc. Designing Privacy Enhancing Technologies: Workshop on Design Issues in Anonymity and Unobservability*. Springer-Verlag Berlin Heidelberg, July 2000. (Cited on page 19)

[113] Roger Dingledine, Nick Mathewson, and Paul F. Syverson. Tor: The Second-Generation Onion Router. In *Proc. 13th USENIX Security Symposium (USENIX 2004)*, volume 13 of *SSYM'04*, pages 303–320. USENIX Association, 2004. (Cited on pages 19, 24, 96, and 104)

[114] Roger Dingledine, Vitaly Shmatikov, and Paul F. Syverson. Synchronous Batching: From Cascades to Free Routes. In *Proc. Privacy Enhancing Technologies Workshop (PET 2004)*, volume 3424 of *Selected Notes in Computer Science*, pages 186–206. Springer-Verlag Berlin Heidelberg, 2004. (Cited on page 154)

[115] Danny Dolev, Ruediger Reischuk, and H. Raymond Strong. Early Stopping in Byzantine Agreement. *Journal of the ACM (JACM)*, 37(4):720–741, October 1990. (Cited on page 145)

[116] J. Domingo-Ferrer and V. Torra. A Critique of k-Anonymity and Some of Its Enhancements. In *Proc. 2008 Third International Conference on Availability, Reliability and Security*, pages 990–993, March 2008. (Cited on page 21)

[117] Charles Duhigg. *The Power Of Habit: Why We Do What We Do In Life And Business.* Random House, New York City, New York, 2012. (Cited on page 2)

[118] George T. Duncan and Sumitra Mukherjee. Optimal Disclosure Limitation Strategy in Statistical Databases: Deterring Tracker Attacks Through Additive Noise. *Journal of the American Statistical Association*, 95(451):720–729, 2000. (Cited on page 20)

[119] Cynthia Dwork. Differential Privacy. In *Proc. Automata, Languages and Programming, 33rd International Colloquium (ICALP 2006)*, volume 4052 of *Selected Notes in Computer Science*, pages 1–12. Springer-Verlag Berlin Heidelberg, 2006. (Cited on pages 15 and 21)

[120] Kevin P. Dyer, Scott E. Coull, and Thomas Shrimpton. Marionette: A Programmable Network-traffic Obfuscation System. In *Proc. 24th USENIX Security Symposium (USENIX))*, SEC'15, pages 367–382, Berkeley, CA, USA, 2015. USENIX Association. (Cited on page 20)

[121] M. Edman, F. Sivrikaya, and B. Yener. A Combinatorial Approach to Measuring Anonymity. In *Proc. 2007 IEEE Intelligence and Security Informatics*, pages 356–363, May 2007. (Cited on page 15)

[122] Harold M. Edwards. A Normal Form for Elliptic Curves. *Bulletin of the American Mathematical Society*, 44:393–422, 2007. (Cited on page 99)

[123] Kaoutar Elkhiyaoui, Erik-Oliver Blass, and Refik Molva. T-Match: Privacy-Preserving Item Matching for Storage-Only RFID Tags. In *Proc. Workshop on RFID Security and Privacy (RFIDSec 2013)*, volume 7739 of *Selected Notes in Computer Science*, pages 76–95. Springer-Verlag Berlin Heidelberg, 2013. (Cited on pages 55 and 68)

[124] D. Ellard, C. Jones, V. Manfredi, W. T. Strayer, B. Thapa, M. Van Welie, and A. Jackson. Rebound: Decoy Routing on Asymmetric Routes via Error Mes-

sages. In *Proc. IEEE 40th Conference on Local Computer Networks (LCN 2015)*, pages 91–99, Oct 2015. (Cited on page 20)

[125] Council of the European Union European Parliament. Regulation (EC) No 1272/2008 of the European Parliament and of the Council of 16 December 2008 on Classification, labelling and Packaging of Substances and Mixtures, Amending and Repealing Directives 67/548/EEC and 1999/45/EC, and Amending Regulation (EC) No 1907/2006, 2008. http://new.eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:32008R1272&rid=3. (Cited on page 68)

[126] European Parliament and the European Council. Data Protection Directive 95/46/EC, November 1995 accessed 2017. http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=CELEX:31995L0046:en:HTML. (Cited on page 33)

[127] Nathan S. Evans, Roger Dingledine, and Christian Grothoff. A Practical Congestion Attack on Tor Using Long Paths. In *Proc. 18th USENIX Security Symposium (USENIX 2009)*, pages 33–50, 2009. (Cited on page 25)

[128] Reza Rezaeian Farashahi. Hashing into Hessian Curves. In *Proc. Progress in Cryptology (AFRICACRYPT 2011)*, volume 6737 of *Selected Notes in Computer Science*, pages 278–289. Springer Berlin Heidelberg, 2011. (Cited on page 106)

[129] Reza Rezaeian Farashahi, Pierre-Alain Fouque, Igor Shparlinski, Mehdi Tibouchi, and José Felipe Voloch. Indifferentiable Deterministic Hashing to Elliptic and Hyperelliptic Curves. *Mathematics of Computation*, 82(281), 2013. (Cited on page 106)

[130] Jean-Charles Faugère, Ludovic Perret, Christophe Petit, and Guénaël Renault. Improving the Complexity of Index Calculus Algorithms in Elliptic Curves over Binary Fields. In *Proc. Advances in Cryptology (EUROCRYPT 2012)*, volume 7237 of *Selected Notes in Computer Science*, pages 27–44. Springer Berlin Heidelberg, 2012. (Cited on page 125)

[131] Nick Feamster, Magdalena Balazinska, Greg Harfst, Hari Balakrishnan, and David Karger. Infranet: Circumventing Web Censorship and Surveillance. In *Proc. 11th USENIX Security Symposium (USENIX)*, August 2002. (Cited on page 20)

[132] Stephen E. Fienberg and Julie McIntyre. Data Swapping: Variations on a Theme by Dalenius and Reiss. In *Proc. Privacy in Statistical Databases (PSD 2004)*, volume 3050 of *Selected Notes in Computer Science*, pages 14–29. Springer-Verlag Berlin Heidelberg, 2004. (Cited on page 20)

[133] Pierre-Alain Fouque, Antoine Joux, and Mehdi Tibouchi. Injective Encodings to Elliptic Curves. In *Information Security and Privacy: 18th Australasian Conference (ACISP 2013)*, volume 7959 of *Selected Notes in Computer Science*, pages 203–218. Springer Berlin Heidelberg, 2013. (Cited on pages 106, 108, 111, 115, and 133)

[134] Pierre-Alain Fouque, Reynald Lercier, Denis Réal, and Frédéric Valette. Fault Attack on Elliptic Curve Montgomery Ladder Implementation. In *Proc. Fifth International Workshop on Fault Diagnosis and Tolerance in Cryptography (FDTC 2008)*, pages 92–98. IEEE Press., 2008. (Cited on page 127)

[135] Pierre-Alain Fouque and Mehdi Tibouchi. Deterministic Encoding and Hashing to Odd Hyperelliptic Curves. In *Proc. Pairing-Based Cryptography (Pairing 2010)*, volume 6487 of *Selected Notes in Computer Science*, pages 265–277. Springer Berlin Heidelberg, 2010. (Cited on page 106)

[136] Pierre-Alain Fouque and Mehdi Tibouchi. Estimating the Size of the Image of Deterministic Hash Functions to Elliptic Curves. In *Proc. Progress in Cryptology (LATINCRYPT 2010)*, volume 6212 of *Selected Notes in Computer Science*, pages 81–91. Springer Berlin Heidelberg, 2010. (Cited on page 106)

[137] Pierre-Alain Fouque and Mehdi Tibouchi. Indifferentiable Hashing to Barreto-Naehrig Curves. In *Proc. Progress in Cryptology (LATINCRYPT 2012)*, volume 7533 of *Selected Notes in Computer Science*, pages 1–17. Springer Berlin Heidelberg, 2012. (Cited on page 106)

[138] Christian Franck. New Directions for Dining Cryptographers. Master's thesis, University of Luxembourg, 2008. (Cited on pages 25, 74, and 75)

[139] Christian Franck. Dining Cryptographers with 0.924 Verifiable Collision Resolution. *Annales UMCS, Informatica*, 14(1):49–59, 2014. (Cited on pages 75 and 91)

[140] Gerhard Frey. How to Disguise an Elliptic Curve (Weil Descent). website, 1998 accessed 2017. http://cacr.uwaterloo.ca/conferences/1998/ecc98/frey.ps. (Cited on page 125)

[141] Franz Fürbass and Johannes Wolkerstorfer. ECC Processor with Low Die Size for RFID Applications. In *Proc. IEEE International Symposium on Circuits and Systems (ISCAS 2007)*, pages 1835–1838. IEEE Press., 2007. (Cited on page 68)

[142] Steven D. Galbraith, John Malone-Lee, and Nigel P. Smart. Public Key Signatures in the Multi-user Setting. *Information Processing Letters*, 83(5):263–266, 2002. (Cited on page 112)

[143] Christina Garman, Matthew Green, Ian Miers, and Aviel D Rubin. Rational Zero: Economic Security for Zerocoin with Everlasting Anonymity. In *Proc. International Conference on Financial Cryptography and Data Security*, pages 140–155. Springer-Verlag Berlin Heidelberg, 2014. (Cited on page 19)

[144] Pierrick Gaudry and Emmanuel Thomé. The mpFq Library and Implementing Curve-based Key Exchanges. In *Software Performance Enhancement for Encryption and Decryption (SPEED 2007)*, pages 49–64, 2007. (Cited on page 127)

[145] Craig Gentry. Fully Homomorphic Encryption Using Ideal Lattices. In *Proc. Forty-first Annual ACM Symposium on Theory of Computing*, STOC '09, pages 169–178, New York, NY, USA, 2009. ACM. (Cited on page 21)

[146] Gianmarco Baldini, Trevor Peirce, Maarten Botterman *et al.* IoT Governance, Privacy and Security Issues. Position paper, European Research Cluster on the Internet of Things, 2015. (Cited on page 34)

[147] Benedikt Gierlichs, Carmela Troncoso, Claudia Diaz, Bart Preneel, and Ingrid Verbauwhede. Revisiting a Combinatorial Approach Toward Measuring Anonymity. In *Proc. 7th ACM Workshop on Privacy in the Electronic Society (WPES)*, pages 111–116, New York, NY, USA, 2008. ACM. (Cited on page 15)

[148] Sharad Goel, Mark Robson, Milo Polte, and Emin Gun Sirer. Herbivore: A Scalable and Efficient Protocol for Anonymous Communication. Technical Report 2003-1890, Cornell University, 2003. (Cited on pages 25, 73, and 88)

[149] I. Goldberg. Improving the Robustness of Private Information Retrieval. In *Proc. 28th IEEE Symposium on Security and Privacy (SP 2007)*, pages 131–148, 2007. (Cited on page 19)

[150] Ian Goldberg, Douglas Stebila, and Berkant Ustaoglu. Anonymity and One-way Authentication in Key Exchange Protocols. *Designs, Codes and Cryptography*, 67(2):245–269, 2013. (Cited on pages 24 and 105)

[151] D. M. Goldschlag, M. G. Reed, and P. F. Syverson. Onion Routing. *Communications of the ACM*, 42(2):39–41, 1999. (Cited on pages 18, 20, 21, and 24)

[152] Shafi Goldwasser and Mihir Bellare. Lecture Notes on Cryptography. website, 2008. http://cseweb.ucsd.edu/users/mihir/papers/gb.pdf. (Cited on page 51)

[153] Shafi Goldwasser and Silvio Micali. Probabilistic Encryption; How to Play Mental Poker Keeping Secret All Partial Information. In *Proc. Fourteenth Annual ACM Symposium on Theory of Computing (STOC 1982)*, pages 365–377. ACM, 1982. (Cited on page 47)

[154] Shafi Goldwasser, Silvio Micali, and Ronald L. Rivest. A Digital Signature Scheme Secure Against Adaptive Chosen-Message Attacks, 1995. (Cited on page 145)

[155] Philippe Golle, Markus Jakobsson, Ari Juels, and Paul F. Syverson. Universal Re-encryption for Mixnets. In *Proc. Topics in Cryptology (CT-RSA 2004)*, volume 2964 of *Selected Notes in Computer Science*, pages 163–178. Springer-Verlag Berlin Heidelberg, 2004. (Cited on page 24)

[156] Philippe Golle and Ari Juels. Dining Cryptographers Revisited. In Christian Cachin and Jan L. Camenisch, editors, *Proc. Advances in Cryptology (EUROCRYPT 2004)*, volume 3027 of *Selected Notes in Computer Science*, pages 456–473. Springer Berlin Heidelberg, 2004. (Cited on pages 25, 75, and 91)

[157] Dan Goodin. Powerful Backdoor/Rootkit Found Preinstalled on 3 million Android Phones. website, 2016 (accessed January 2017). https://arstechnica.com/security/2016/11/powerful-backdoorrootkit-found-preinstalled-on-3-million-android-phones/. (Cited on page 4)

[158] Google Company. Our History in Depth. website, accessed 2017. https://www.google.com/about/company/history/. (Cited on page 2)

[159] Marcin Gorawski, Zacheusz Siedlecki, and Anna Gorawska. Collaborative Multiparty Association Rules Mining with Threshold Homomorphic Encryption. In *Proc. ICA3PP International Workshops and Symposiums on Algorithms and Architectures for Parallel Processing - Volume 9532*, pages 251–263, New York, NY, USA, 2015. Springer-Verlag New York. (Cited on page 21)

[160] Seda Gürses and Bettina Berendt. *PETs in the Surveillance Society: A Critical Review of the Potentials and Limitations of the Privacy as Confidentiality Paradigm*, pages 301–321. Springer Netherlands, Dordrecht, 2010. (Cited on page 6)

[161] Shai Halevi and Shai Micali. Practical and Provably-Secure Commitment Schemes from Collision-Free Hashing. In *Proc. Advances in Cryptology (CRYPTO 1996)*, volume 1109 of *Selected Notes in Computer Science*, pages 201–215. Springer Berlin Heidelberg, 1996. (Cited on page 145)

[162] Uwe Hansmann, Lothar Merk, Martin S. Nicklous, and Thomas Stober. *Pervasive Computing: The Mobile World*. Springer-Verlag Berlin Heidelberg, Berlin, Germany, 2nd edition, August 2003. (Cited on page 29)

[163] Daniel Hein, Johannes Wolkerstorfer, and Norbert Felber. ECC Is Ready for RFID – A Proof in Silicon. In *Proc. Selected Areas in Cryptography: 15th International Workshop (SAC 2008)*, volume 5381 of *Selected Notes in Computer Science*, pages 401–413. Springer-Verlag Berlin Heidelberg, 2008. (Cited on page 68)

[164] Matthew A. Henry. Comments Of Golden Frog. website, 2014 accessed 2017. https://s3.amazonaws.com/s3.documentcloud.org/documents/1312218/golden-frog-comments-fcc-gn-14-28-final.pdf. (Cited on page 95)

[165] Jens Hermans, Andreas Pashalidis, Frederik Vercauteren, and Bart Preneel. A New RFID Privacy Model. In *Proc. Computer Security (ESORICS 2011)*, volume 6879 of *Selected Notes in Computer Science*, pages 568–587. Springer-Verlag Berlin Heidelberg, 2011. (Cited on page 53)

[166] Dominik Herrmann, Rolf Wendolsky, and Hannes Federrath. Website Fingerprinting: Attacking Popular Privacy Enhancing Technologies with the Multinomial NaïVe-bayes Classifier. In *Proc. ACM Workshop on Cloud Computing Security (CCSW 2009)*, pages 31–42, New York, NY, USA, 2009. ACM. (Cited on page 25)

[167] Thomas S. Heydt-Benjamin, Hee-Jin Chae, Benessa Defend, and Kevin Fu. Privacy for Public Transportation. In *Proc. Privacy Enhancing Technologies Workshop (PET 2006)*, pages 1–19, Berlin, Heidelberg, 2006. Springer-Verlag Berlin Heidelberg. (Cited on page 18)

[168] Mireille Hildebrandt. Privacy and Identity. *Privacy and the Criminal Law*, 43:61–104, 2006. (Cited on page 14)

[169] Mireille Hildebrandt and Bert-Jaap Koops. The Challenges of Ambient Law and Legal Protection in the Profiling Era. *The Modern Law Review*, 73(3):428–460, 2010. (Cited on page 4)

[170] Gesine Hinterwälder, Felix Riek, and Christof Paar. Efficient E-cash with Attributes on MULTOS Smartcards. In *Proc. Workshop on RFID Security and Privacy (RFIDsec 2015)*, pages 141–155, New York, NY, USA, 2015. Springer-Verlag New York. (Cited on page 18)

[171] Hüseyin Hisil, Kenneth Koon-Ho Wong, Gary Carter, and Ed Dawson. Twisted Edwards Curves Revisited. In *Proc. Advances in Cryptology (ASIACRYPT 2008)*, volume 5350 of *Selected Notes in Computer Science*, pages 326–343. Springer Berlin Heidelberg, 2008. (Cited on page 128)

[172] Hitachi Develops a New RFID with Embedded Antenna μ-Chip. website, 2003 accessed 2017s. http://www.hitachi.com/New/cnews/030902.html. (Cited on pages 31 and 45)

[173] Amir Houmansadr, Giang T. K. Nguyen, Matthew Caesar, and Nikita Borisov. Cirripede: Circumvention Infrastructure using Router Redirection with Plausible Deniability. In *Proc. 18th ACM Conference on Computer and Communications Security (CCS 2011)*, October 2011. (Cited on page 20)

[174] Amir Houmansadr, Thomas Riedl, Nikita Borisov, and Andrew Singer. I Want my Voice to be Heard: IP over Voice-over-IP for Unobservable Censorship Circumvention. In *Proc. Network and Distributed System Security Symposium (NDSS 2013)*. Internet Society, February 2013. (Cited on page 20)

[175] Maged H Ibrahim. SecureCoin: A Robust Secure and Efficient Protocol for Anonymous Bitcoin Ecosystem. *International Journal of Network Security*, 19(2):295–312, 2017. (Cited on page 19)

[176] Thomas Icart. How to Hash into Elliptic Curves. In *Proc. Advances in Cryptology (CRYPTO 2009)*, volume 5677 of *Selected Notes in Computer Science*, pages 303–316. Springer Berlin Heidelberg, 2009. (Cited on page 106)

[177] The OpenNet Initiative. Filtering Data. website, 2013 accessed 2017. https://opennet.net/research/data. (Cited on pages 14 and 95)

[178] Institute of Electrical and Electronics Engineers. *P1363 Draft Standard Specifications for Public Key Cryptography*. IEEE Press., 1999. (Cited on page 126)

[179] IBM Internet of Things Foundation. web-site, accessed January 2017. https://internetofthings.ibmcloud.com/. (Cited on page 31)

[180] Malika Izabachène and Benoît Libert. Divisible e-Cash in the Standard Model. In *Proc. 5th International Conference on Pairing-Based Cryptography*, Pairing'12, pages 314–332, Berlin, Heidelberg, 2013. Springer-Verlag Berlin Heidelberg. (Cited on page 18)

[181] Tetsuya Izu and Tsuyoshi Takagi. Exceptional Procedure Attack on Elliptic Curve Cryptosystems. In *Proc. Public Key Cryptography (PKC 2003)*, volume 2567 of *Selected Notes in Computer Science*, pages 224–239. Springer Berlin Heidelberg, 2003. (Cited on page 127)

[182] Markus Jakobsson. Flash Mixing. In *Proc. Eighteenth Annual ACM Symposium on Principles of Distributed Computing*, PODC '99, pages 83–89, New York, NY, USA, 1999. ACM. (Cited on page 24)

[183] Markus Jakobsson and Ari Juels. An Optimally Robust Hybrid Mix Network. In *Proc. Twentieth Annual ACM Symposium on Principles of Distributed Computing*, PODC '01, pages 284–292, New York, NY, USA, 2001. ACM. (Cited on pages 23, 145, and 157)

[184] Rob Jansen, Florian Tschorsch, Aaron Johnson, and Björn Scheuermann. The Sniper Attack: Anonymously Deanonymizing and Disabling the Tor Network. In *NDSS 2014)*, 2014. (Cited on page 25)

[185] Burton Stephen Kaliski Jr. A Pseudo-Random Bit Generator Based on Elliptic Logarithms. In *Proc. Advances in Cryptology (CRYPTO 1986)*, volume 263 of *Selected Notes in Computer Science*, pages 84–103. Springer Berlin Heidelberg, 1986. (Cited on page 106)

[186] Burton Stephen Kaliski Jr. *Elliptic Curves and Cryptography: A Pseudorandom Bit Generator and Other Tools*. PhD thesis, MIT, 1988. http://theory.lcs.mit.edu/~cis/theses/kaliski-phd.pdf. (Cited on page 106)

[187] Ari Juels. "Yoking-proofs" for RFID Tags. In *Proc. Second IEEE Annual Conference on Pervasive Computing and Communications Workshops (PerCom 2004)*, pages 138–143. IEEE Press., 2004. (Cited on page 68)

[188] Ari Juels and Stephen A. Weis. Defining Strong Privacy for RFID. In *Proc. Fifth Annual IEEE International Conference on Pervasive Computing and Communica-*

tions Workshops - PerCom 2007, pages 342–347. IEEE Press., 2007. (Cited on page 53)

[189] Josh Karlin, Daniel Ellard, Alden W. Jackson, Christine E. Jones, Greg Lauer, David P. Mankins, and W. Timothy Strayer. Decoy Routing: Toward Unblockable Internet Communication. In *Proc. USENIX Workshop on Free and Open Communications on the Internet (FOCI 2011)*, August 2011. (Cited on page 20)

[190] A. Kate and I. Goldberg. Using Sphinx to Improve Onion Routing Circuit Construction. In *Proc. 14th Conference on Financial Cryptography and Data Security (FC 2010)*, volume 6052 of *Selected Notes in Computer Science*, pages 359–366. Springer Berlin Heidelberg, 2010. (Cited on page 24)

[191] A. Kate, G. M. Zaverucha, and I. Goldberg. Pairing-Based Onion Routing with Improved Forward Secrecy. *ACM Transactions on Information and System Security (TISSEC)*, 13(4):29, 2010. (Cited on page 24)

[192] Anne V. Kayem, C. T. Vester, and Christoph Meinel. Automated k-Anonymization and l-Diversity for Shared Data Privacy. In *Proc. 27th International Conference on Database and Expert Systems Applications - Volume 9827*, DEXA 2016, pages 105–120, New York, NY, USA, 2016. Springer-Verlag New York. (Cited on page 21)

[193] Sheharbano Khattak, Laurent Simon, and Steven J. Murdoch. Systemization of Pluggable Transports for Censorship Resistance. *CoRR*, abs/1412.7448, 2014. (Cited on page 20)

[194] Rachel King. Samsung at CES 2015: Internet-of-Things is not Science Diction, but 'Science Fact'. website, 2015 (accessed January 2017). http://www.zdnet.com/article/ces-2015-samsung-internet-of-things/. (Cited on page 30)

[195] L. Kissner, A. Oprea, M. Reiter, D. Song, and K. Yang. Private Keyword-based Push and Pull with Applications to Anonymous Communication. *Applied Cryptography and Network Security*, 2004. (Cited on page 19)

[196] Neal Koblitz. Elliptic Curve Cryptosystems. *Mathematics of Computation*, 48(177):203–209, 1987. (Cited on page 97)

[197] Neal Koblitz and Alfred Menezes. Pairing-Based Cryptography at High Security Levels. In *Proc. 10th IMA Conference on Cryptography and Coding, 2005*, vol-

ume 3796 of *Selected Notes in Computer Science*, pages 13–36. Springer Berlin Heidelberg, 2005. (Cited on page 100)

[198] Neal Koblitz and Alfred Menezes. The Random Oracle Model: a Twenty-year Retrospective. *Designs, Codes and Cryptography*, 77(2):587–610, 2015. (Cited on page 50)

[199] Neal Koblitz and J. Alfred Menezes. Another Look at "Provable Security". *Journal of Cryptology*, 20(1):3–37, 2005. (Cited on page 50)

[200] Jacob Kohnstamm and Drudeisha Madhub. Mauritius Declaration on the Internet of Things. In *Proc. 36th International Conference of Data Protection and Privacy Commissioners, 2014*, 2014. (Cited on page 34)

[201] Stefan Köpsell and Ulf Hilling. How to Achieve Blocking Resistance for Existing Systems Enabling Anonymous Web Surfing. In *Proc. ACM Workshop on Privacy in the Electronic Society (WPES 2004)*, October 2004. (Cited on page 20)

[202] Anna Krasnova, Moritz Neikes, and Peter Schwabe. *Footprint Scheduling for Dining-Cryptographer Networks*, pages 385–402. Springer Berlin Heidelberg, 2017. (Cited on page 11)

[203] Dennis Kügler. An Analysis of GNUnet and the Implications for Anonymous, Censorship-Resistant Networks. In Roger Dingledine, editor, *Proc. Privacy Enhancing Technologies Workshop (PET 2003)*, pages 161–176. Springer-Verlag Berlin Heidelberg, March 2003. (Cited on page 19)

[204] Eyal Kushilevitz and Rafail Ostrovsky. Replication is NOT Needed: SINGLE Database, Computationally-Private Information Retrieval. In *Proc. 38th Annual Symposium on Foundations of Computer Science (FOCS 1997)*, pages 364–373, 1997. (Cited on page 19)

[205] Albert Kwon, David Lazar, Srinivas Devadas, and Bryan Ford. Riffle: An Efficient Communication System With Strong Anonymity. *PoPETs*, 2016(2):115–134, 2016. (Cited on pages 20 and 23)

[206] Laurie E. Law and Jerome A. Solinas. Suite B Cryptographic Suites for IPsec. website, 2011 accessed 2017. https://tools.ietf.org/html/rfc6379. (Cited on page 125)

[207] Michael Z. Lee, Alan M. Dunn, Jonathan Katz, Brent Waters, and Emmett Witchel. Anon-Pass: Practical Anonymous Subscriptions. In *Proc. 34th IEEE*

Symposium on Security and Privacy (SP 2013), pages 319–333. IEEE Press., May 2013. (Cited on page 19)

[208] Yong Ki Lee, Kazuo Sakiyama, Lejla Batina, and Ingrid Verbauwhede. Elliptic-Curve-Based Security Processor for RFID. *IEEE Transactions on Computers*, 57(11):1514–1527, 2008. (Cited on page 68)

[209] Arjen K. Lenstra, Thorsten Kleinjung, and Emmanuel Thomé. Universal Security - From Bits and Mips to Pools, Lakes - and Beyond. In *Number Theory and Cryptography - Papers in Honor of Johannes Buchmann on the Occasion of His 60th Birthday*, volume 8260 of *Selected Notes in Computer Science*, pages 121–124. Springer Berlin Heidelberg, 2013. (Cited on page 47)

[210] Jing Li, Xiong Li, Licheng Wang, Debiao He, and Xinxin Niu. Oblivious Transfer Protocols Based on Group Factoring Problem. In Leonard Barolli, Fatos Xhafa, and Kangbin Yim, editors, *Proc. 11th International Conference On Broad-Band Wireless Computing, Communication and Applications (BWCCA 2016)*, pages 885–892. Springer International Publishing, 2016. (Cited on page 19)

[211] Ninghui Li, Tiancheng Li, and Suresh Venkatasubramanian. t-Closeness: Privacy Beyond k-Anonymity and l-Diversity. In *Proc. 23rd International Conference on Data Engineering (ICDE 2007)*, pages 106–115. IEEE Press., 2007. (Cited on page 21)

[212] Shuai Li, Mike Schliep, and Nick Hopper. Facet: Streaming over Videoconferencing for Censorship Circumvention. In *Proc. 12th ACM Workshop on Privacy in the Electronic Society (WPES)*, November 2014. (Cited on page 20)

[213] Yibin Li, Wenyun Dai, Zhong Ming, and Meikang Qiu. Privacy Protection for Preventing Data Over-Collection in Smart City. *IEEE Transactions on Computers*, PP(99):1–11, 2015. (Cited on pages 36 and 136)

[214] Rudolf Lidl and Harald Niederreiter. *Finite Fields*, volume 20 of *Encyclopedia of Mathematics and its Applications*. Cambridge University Press, 1997. (Cited on page 116)

[215] Yehuda Lindell and Benny Pinkas. Privacy Preserving Data Mining. In *Proc. Advances in Cryptology (CRYPTO 2000)*, CRYPTO '00, pages 36–54, London, UK, UK, 2000. Springer-Verlag Berlin Heidelberg. (Cited on pages 20 and 21)

[216] Logitech Pop. website, accessed January 2017. http://www.logitech.com/en-us/product/12579. (Cited on pages 4 and 30)

[217] Julio López and Ricardo Dahab. Fast Multiplication on Elliptic Curves over GF(2^m) without Precomputation. In *Proc. Cryptographic Hardware and Embedded Systems (CHES 1999)*, volume 1717 of *Selected Notes in Computer Science*, pages 316–327. Springer Berlin Heidelberg, 1999. (Cited on page 103)

[218] Ashwin Machanavajjhala, Daniel Kifer, Johannes Gehrke, and Muthuramakrishnan Venkitasubramaniam. L-diversity: Privacy Beyond K-anonymity. *ACM Trans. Knowl. Discov. Data*, 1(1), March 2007. (Cited on page 21)

[219] Jeremy Malcolm. Australian Copyright Censorship Bill Could Block VPNs and Circumvention Information. website, 2015 accessed 2017. https://www.eff.org/deeplinks/2015/04/australian-copyright-censorship-bill-could-block-vpns-and-circumvention. (Cited on page 95)

[220] M. B. Malik, M. A. Ghazi, and R. Ali. Privacy Preserving Data Mining Techniques: Current Scenario and Future Prospects. In *Proc. 2012 Third International Conference on Computer and Communication Technology*, pages 26–32, Nov 2012. (Cited on page 20)

[221] Alexander J. Martin. France's 3-month State of Emergency Lets Govt Censor the Web. website, 2015 accessed 2017. http://www.theregister.co.uk/2015/11/20/france_extends_state_emergency_3_months_police_censor_web/. (Cited on page 95)

[222] Friedemann Mattern and Christian Floerkemeier. *From the Internet of Computers to the Internet of Things*, volume 6462 of *Selected Notes in Computer Science*, pages 242–259. Springer Berlin Heidelberg, 2010. (Cited on page 34)

[223] Gregory Maxwell. CoinJoin: Bitcoin Privacy for the Real World (2013). https://bitcointalk.org/index.php. (Cited on page 19)

[224] D. McClure and J. P. Reiter. Assessing Disclosure Risks for Synthetic Data with Arbitrary Intruder Knowledge. *Statistical Journal of the International Association of Official Statistics*, 32(1):109–126, 2016. (Cited on page 20)

[225] Carlos AGUILAR MELCHOR and Philippe GABORIT. A Lattice-Based Computationally-Efficient Private Information Retrieval Protocol, 2007. Short version presented in WEWORC, in July 2007, Bochum, Germany carlos.aguilar@unilim.fr 13844 received 27 Nov 2007, last revised 27 Nov 2007. (Cited on page 19)

[226] Ralph C. Merkle. One Way Hash Functions and DES. In *Proc. Advances in Cryptology (CRYPTO 1989)*, volume 435 of *Selected Notes in Computer Science*, pages 428–446. Springer-Verlag Berlin Heidelberg, 1990. (Cited on page 50)

[227] Victor S. Miller. Use of Elliptic Curves in Cryptography. In *Proc. Advances in Cryptology (CRYPTO 1985)*, volume 218 of *Selected Notes in Computer Science*, pages 417–426. Springer Berlin Heidelberg, 1985. (Cited on page 97)

[228] Bodo Möller. A Public-Key Encryption Scheme with Pseudo-random Ciphertexts. In *Proc. 9th European Symposium on Research in Computer Security (ESORICS 2004)*, volume 3193 of *Selected Notes in Computer Science*, pages 335–351. Springer Berlin Heidelberg, 2004. (Cited on page 103)

[229] Peter L. Montgomery. Speeding the Pollard and Elliptic Curve Methods of Factorization. *Mathematics of Computation*, 48(177):243–264, 1987. (Cited on pages 99 and 103)

[230] Glyn Moody. Russia Prepares To Block Tor And Anonymizing Proxies. website, 2013 accessed 2017. https://www.techdirt.com/articles/20130820/09464424253/russia-prepares-to-block-tor-anonymizing-proxies.shtml. (Cited on page 96)

[231] Wojciech Mostowski and Pim Vullers. Efficient U-Prove Implementation for Anonymous Credentials on Smart Cards. In *Security and Privacy in Communication Networks (SecureComm 2011)*, volume 96 of *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering (LNICST)*, pages 243–260. Springer Berlin Heidelberg, 2011. (Cited on page 35)

[232] Steven J. Murdoch and George Danezis. Low-Cost Traffic Analysis of Tor. In *Proc. 26th IEEE Symposium on Security and Privacy (SP 2005)*, pages 183–195. IEEE Press., 2005. (Cited on pages 25 and 71)

[233] Satoshi Nakamoto. Bitcoin: A Peer-to-peer Electronic Cash System, 2008. http://www.bitcoin.org/bitcoin.pdf. (Cited on page 19)

[234] Arvind Narayanan and Vitaly Shmatikov. Robust De-anonymization of Large Sparse Datasets. In *Proc. 29th IEEE Symposium on Security and Privacy (SP 2008)*, pages 111–125. IEEE Press., 2008. (Cited on page 34)

[235] M. Narwaria and S. Arya. Privacy Preserving Data Mining – 'A State of the Art'. In *Proc. 3rd International Conference on Computing for Sustainable Global Development (INDIACom 2016)*, pages 2108–2112, March 2016. (Cited on page 20)

[236] C. Andrew Neff. A Verifiable Secret Shuffle and Its Application to e-Voting. In *Proc. 8th ACM Conference on Computer and Communications Security (CCS 2001)*, pages 116–125. ACM, 2001. (Cited on page 23)

[237] Nest Labs. website, accessed January 2017. https://nest.com. (Cited on page 30)

[238] Jason Q. Ng. Reputation Matters: Unpacking the Microsoft China Censorship Scandal. website, 2014 accessed 2017. http://blogs.wsj.com/chinarealtime/2014/02/14/reputation-matters-unpacking-the-microsoft-china-censorship-scandal/. (Cited on pages 14 and 95)

[239] Jesper Buus Nielsen. Separating Random Oracle Proofs from Complexity Theoretic Proofs: The Non-committing Encryption Case. In *Proc. Advances in Cryptology (CRYPTO 2002)*, volume 2442 of *Selected Notes in Computer Science*, pages 111–126. Springer Berlin Heidelberg, 2002. (Cited on page 50)

[240] Eva Nieuwdorp. The Pervasive Discourse: An Analysis. *Comput. Entertain.*, 5(2), April 2007. (Cited on page 29)

[241] NIST. Digital Signature Standard. Standard 186-2, National Institute for Standards and Technology, 2000. http://csrc.nist.gov/publications/fips/archive/fips186-2/fips186-2.pdf. (Cited on page 98)

[242] Miyako Ohkubo and Masayuki Abe. A Length-Invariant Hybrid Mix. In Tatsuaki Okamoto, editor, *Proc. Advances in Cryptology (ASIACRYPT 2000)*, volume 1976 of *Selected Notes in Computer Science*, pages 178–191. Springer-Verlag Berlin Heidelberg, 2000. (Cited on pages 23, 145, and 157)

[243] Thomaz Oliveira, Julio López, Diego F. Aranha, and Francisco Rodríguez-Henríquez. Lambda Coordinates for Binary Elliptic Curves. In *Proc. Cryptographic Hardware and Embedded Systems (CHES 2013)*, volume 8086 of *Selected Notes in Computer Science*, pages 311–330. Springer Berlin Heidelberg, 2013. (Cited on page 125)

[244] OASIS Open. Profiles for the OASIS Security Assertion Markup Language (SAML). Standard, OASIS Open, 2005. (Cited on page 35)

[245] L. Øverlier and P. Syverson. Improving Efficiency and Simplicity of Tor Circuit Establishment and Hidden Services. In *Proc. Privacy Enhancing Technologies Symposium (PETS 2007)*, pages 134–152, 2007. (Cited on page 24)

[246] L. Øverlier and P. F. Syverson. Locating Hidden Servers. In *Proc. 27th IEEE Symposium on Security and Privacy (SP 2006)*, pages 100–114, 2006. (Cited on pages 18, 20, and 24)

[247] Kiran P and Kavya N.p. Article: A Survey on Methods, Attacks and Metric for Privacy Preserving Data Publishing. *International Journal of Computer Applications*, 53(18):20–28, September 2012. (Cited on page 20)

[248] Christian Paquin. U-Prove Technology Overview V1.1. Technical report, Microsoft Research, 2013. (rev 2). (Cited on pages 35 and 37)

[249] Eli Pariser. *The Filter Bubble: What the Internet Is Hiding from You*. Penguin Group , The, 2011. (Cited on page 14)

[250] Choonsik Park, Kazutomo Itoh, and Kaoru Kurosawa. Efficient Anonymous Channel and All/Nothing Election Scheme. In *Proc. Advances in Cryptology: Workshop on the Theory and Application of Cryptographic Techniques (EUROCRYPT 1993)*, volume 765 of *Selected Notes in Computer Science*, pages 248–259. Springer, 1993. (Cited on pages 24, 145, and 157)

[251] Fernando Patolsky and Charles M. Lieber. Nanowire Nanosensors. *Materials Today*, 8(4):20 – 28, 2005. (Cited on page 31)

[252] Daniel Payne. Categories of Websites Blocked by UK Universities. website, 2014 accessed 2017. https://figshare.com/articles/Categories_of_websites_blocked_by_UK_universities/1106875. (Cited on page 95)

[253] Pebble. website, accessed January 2017. https://www.pebble.com/. (Cited on page 30)

[254] G. Bertoni, J. Daemen, M. Peeters and G. Van Assche. The Keccak Reference. Round 3 submission to NIST SHA-3, 2011. (Cited on page 51)

[255] Christophe Petit and Jean-Jacques Quisquater. On Polynomial Systems Arising from a Weil Descent. In *Proc. Advances in Cryptology (ASIACRYPT 2012)*, volume 7658 of *Selected Notes in Computer Science*, pages 451–466. Springer Berlin Heidelberg, 2012. (Cited on page 125)

[256] A Pfitzmann and M Waidner. Networks Without User Observability – Design Options. In *Proc. Advances in Cryptology (EUROCRYPT 1985)*, pages 245–253, New York, NY, USA, 1986. Springer-Verlag New York. (Cited on page 23)

[257] Andreas Pfitzmann. How to Implement ISDNs Without User Observability — Some Remarks. Technical report, Department of Computer Science, University of Karlsruhe, 1985. Internal report 14/85. (Cited on page 73)

[258] Andreas Pfitzmann. Technischer Datenschutz in dienstintegrierenden Digitalnetzen - Problemanalyse, Lösungsansätze und eine angepasste Systemstruktur. In Peter Paul Spies, editor, *Proc. Datenschutz und Datensicherung im Wandel der Informationstechnologien, 1. GI-Fachtagung, 1985*, volume 113 of *Informatik-Fachberichte*, pages 96–112. Springer-Verlag Berlin Heidelberg, 1985. (Cited on pages 18, 20, 22, 23, and 25)

[259] Andreas Pfitzmann. *Diensteintegrierende Kommunikationsnetze mit teilnehmerüberprüfbarem Datenschutz*. PhD thesis, Fakultät für Informatik, Universität Karlsruhe, 1990. (Cited on pages 27, 72, and 74)

[260] Andreas Pfitzmann and Marit Hansen. A Terminology for Talking about Privacy by Data Minimization: Anonymity, Unlinkability, Undetectability, Unobservability, Pseudonymity, and Identity Management. http://dud.inf.tu-dresden.de/literatur/Anon_Terminology_v0.34.pdf, August 2010. v0.34. (Cited on pages 15, 36, and 155)

[261] Andreas Pfitzmann, Birgit Pfitzmann, and Michael Waidner. ISDN-Mixes: Untraceable Communication with Very Small Bandwidth Overhead. In *Kommunikation in verteilten Systemen*, volume 267 of *Informatik-Fachberichte*, pages 451–463. Springer Berlin Heidelberg, 1991. (Cited on pages 18 and 20)

[262] Birgit Pfitzmann. Breaking an Efficient Anonymous Channel. In Alfredo De Santis, editor, *Proc. Advances in Cryptology: Workshop on the Theory and Application of Cryptographic Techniques (EUROCRYPT 1994)*, volume 950 of *Selected Notes in Computer Science*, pages 332–340. Springer-Verlag Berlin Heidelberg, 9–12 May 1994. (Cited on page 147)

[263] Benny Pinkas. Cryptographic Techniques for Privacy-preserving Data Mining. *SIGKDD Explor. Newsl.*, 4(2):12–19, 2002. (Cited on pages 20 and 21)

[264] Generating Business Revenue Growth Through the Use of Internet of Things and Big Data Analytics. website, 2016 (accessed January 2017). http://www.prnewswire.com/news-releases/generating-business-revenue-growth-through-the-use-of-internet-of-things-and-big-data-analytics-300306294.html. (Cited on page 30)

[265] Veena Pureswaran and Paul Brody. Device democracy: Saving the future of the Internet of Things. Technical report, IBM Institute for Business Value, 2015. (Cited on page 32)

[266] Steve Rambam. Talk: Privacy Is Dead âĂŤ Get Over It., 2016 accessed 2017. https://archive.org/details/hope-number-six-privacy-is-dead. (Cited on pages 2, 4, and 5)

[267] Kai Rannenberg, Jan Camenisch, and Ahmad Sabouri, editors. *Attribute-based Credentials for Trust*. Springer International Publishing, 2015. (Cited on pages 33 and 35)

[268] Jean-François Raymond. Traffic Analysis: Protocols, Attacks, Design Issues, and Open Problems. In Hannes Federrath, editor, *Proc. International Workshop on Design Issues in Anonymity and Unobservability*, volume 2009 of *Selected Notes in Computer Science*, pages 10–29. Springer-Verlag Berlin Heidelberg, 2000. (Cited on pages 25, 147, and 155)

[269] M. G. Reed, P. F. Syverson, and D. M. Goldschlag. Anonymous Connections and Onion Routing. *IEEE Journal on Selected Areas in Communications*, 16(4):482–494, 1998. (Cited on pages 71 and 155)

[270] Michael K. Reiter and Aviel D. Rubin. Crowds: Anonymity for Web Transactions. *ACM Trans. Inf. Syst. Secur.*, 1(1):66–92, November 1998. (Cited on page 15)

[271] Quanli; Reiter, Jerome P.; Wang and Biyuan Zhang. Bayesian Estimation of Disclosure Risks for Multiply Imputed, Synthetic Data. *Journal of Privacy and Confidentiality*, 6(1), 2014. (Cited on page 20)

[272] Joost Renes, Craig Costello, and Lejla Batina. Complete Addition Formulas for Prime Order Elliptic Curves. In *Proc. 10th IMA Conference on Cryptography and Coding, 2005*, volume 9665 of *Selected Notes in Computer Science*, pages 403–428. Springer Berlin Heidelberg, 2016. (Cited on page 99)

[273] Janessa Rivera and Rob van der Meulen. Gartner Says the Internet of Things Installed Base Will Grow to 26 Billion Units By 2020. http://www.gartner.com/newsroom/id/2636073, December 2013 accessed 2016. (accessed Dec. 2015). (Cited on pages 32, 34, and 136)

[274] Domenico Rotondi Roberto Minerva, Abyi Biru. Towards a Definition of the Internet of Things (IoT). http://iot.ieee.org/images/files/pdf/IEEE_IoT_Towards_Definition_Internet_of_Things_Revision1_27MAY15.pdf, 2015 (accessed January 2017). Revision 1. (Cited on pages 3, 30, and 31)

[275] Lawrence G. Roberts. ALOHA Packet System with and Without Slots and Capture. *ACM SIGCOMM Computer Communication Review*, 5(2):28–42, 1975. (Cited on page 73)

[276] Phillip Rogaway and Thomas Shrimpton. A Provable-Security Treatment of the Key-Wrap Problem. In *Proc. Advances in Cryptology (EUROCRYPT 2006)*, volume 4004 of *Selected Notes in Computer Science*, pages 373–390. Springer Berlin Heidelberg, 2006. (Cited on page 110)

[277] Stefanie Roos, Benjamin Schiller, Stefan Hacker, and Thorsten Strufe. Measuring Freenet in the Wild: Censorship-resilience under Observation. In *Proc. Privacy Enhancing Technologies Symposium (PETS 2014)*, July 2014. (Cited on page 19)

[278] Karen Rose, Scott Eldridge, and Lyman Chapin. The Internet of Things: An Overview (Understanding the Issues and Challenges of a More Connected World). Technical report, Internet Society, 2015. (Cited on page 32)

[279] Tim Ruffing, Pedro Moreno-Sanchez, and Aniket Kate. CoinShuffle: Practical Decentralized Coin Mixing for Bitcoin. In *Proc. European Symposium on Research in Computer Security*, pages 345–364. Springer-Verlag Berlin Heidelberg, 2014. (Cited on page 19)

[280] Andy Rupp, Foteini Baldimtsi, Gesine Hinterwälder, and Christof Paar. Cryptographic Theory Meets Practice: Efficient and Privacy-Preserving Payments for Public Transport. *ACM Trans. Inf. Syst. Secur.*, 17(3):10:1–10:31, March 2015. (Cited on page 18)

[281] Nazir Saleheen, Supriyo Chakraborty, Nasir Ali, Md Mahbubur Rahman, Syed Monowar Hossain, Rummana Bari, Eugene Buder, Mani Srivastava, and Santosh Kumar. mSieve: Differential Behavioral Privacy in Time Series of Mobile Sensor Data. In *Proc. 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, UbiComp '16, pages 706–717, New York, NY, USA, 2016. ACM. (Cited on page 21)

[282] David Sánchez, Josep Domingo-Ferrer, and Sergio Martínez. *Improving the Utility of Differential Privacy via Univariate Microaggregation*, pages 130–142. Springer International Publishing, 2014. (Cited on page 21)

[283] Marijn Scheir, Josep Balasch, Alfredo Rial, Bart Preneel, and Ingrid Verbauwhede. Anonymous Split E-Cash – Toward Mobile Anonymous Payments. *ACM Trans. Embed. Comput. Syst.*, 14(4):85:1–85:25, September 2015. (Cited on page 18)

[284] Bruce Schneier. China Now Blocking Encryption. website, 2012 accessed 2017. https://www.schneier.com/blog/archives/2012/12/china_now_block.html. (Cited on page 96)

[285] Andrei Serjantov and George Danezis. Towards an Information Theoretic Metric for Anonymity. In *Proc. Privacy Enhancing Technologies Workshop (PET 2002)*, pages 41–53, Berlin, Heidelberg, 2003. Springer-Verlag Berlin Heidelberg. (Cited on page 15)

[286] Andrei Serjantov, Roger Dingledine, and Paul F. Syverson. From a Trickle to a Flood: Active Attacks on Several Mix Types. In *Information Hiding*, volume 2578 of *Selected Notes in Computer Science*, pages 36–52. Springer-Verlag Berlin Heidelberg, 2002. (Cited on page 160)

[287] Andrei Serjantov and Peter Sewell. Passive Attack Analysis for Connection-Based Anonymity Systems. In Einar Snekkenes and Dieter Gollmann, editors, *Proc. Computer Security (ESORICS 2003)*, volume 2808 of *Selected Notes in Computer Science*, pages 116–131. Springer-Verlag Berlin Heidelberg, 2003. (Cited on pages 135 and 155)

[288] Andrew Shallue and Christiaan van de Woestijne. Construction of Rational Points on Elliptic Curves over Finite Fields. In *Proc. Algorithmic Number Theory, 7th International Symposium (ANTS 2006)*, volume 4076 of *Selected Notes in Computer Science*, pages 510–524. Springer Berlin Heidelberg, 2006. (Cited on page 106)

[289] Natalie Shlomo and Ton de Waal. Protection of Micro-data Subject to Edit Constraints Against Statistical Disclosure. *Journal of Official Statistics*, 24(2):229–253, 2008. (Cited on page 20)

[290] Semyon Slepakov. Luba, Youtube Star. website, accessed January 2017. https://www.youtube.com/watch?v=3Qtg8bPgEww. (Cited on page 4)

[291] Nigel Smart, editor. *ECRYPT II Yearly Report on Algorithms and Keysizes*. European Network of Excellence in Cryptology II, 2012. (Cited on page 97)

[292] Daniel J. Solove. *Understanding Privacy*. Harvard University Press, New York, 2010. (Cited on pages 32, 36, and 39)

[293] T. K. Srikanth and Sam Toueg. Simulating Authenticated Broadcasts to Derive Simple Fault-tolerant Algorithms. *Distributed Computing*, 2(2):80–94, 1987. (Cited on page 145)

[294] Chris Studholme and Ian Blake. Multiparty Computation to Generate Secret Permutations. IACR Cryptology ePrint Archive: Report 2007/353, 2007. http://eprint.iacr.org/2007/353. (Cited on page 75)

[295] Mark Sullivan. A Brief History of GPS. website, 2012 accessed 2017. http://www.pcworld.com/article/2000276/a-brief-history-of-gps.html. (Cited on page 2)

[296] Yixin Sun, Anne Edmundson, Laurent Vanbever, Oscar Li, Jennifer Rexford, Mung Chiang, and Prateek Mittal. RAPTOR: Routing Attacks on Privacy in Tor. In *Proc. 24th USENIX Security Symposium (USENIX)*, pages 271–286, 2015. (Cited on page 25)

[297] Latanya Sweeney. Weaving Technology and Policy Together to Maintain Confidentiality. *The Journal of Law, Medicine & Ethics*, 25(2-3):98–110, 1997. (Cited on page 34)

[298] Latanya Sweeney. Uniqueness of Simple Demographics in the U.S. Population. Technical Report LIDAP-WP4, Carnegie Mellon University, School of Computer Science, Data Privacy Laboratory, 2000. (Cited on page 34)

[299] Latanya Sweeney. k-Anonymity: A Model for Protecting Privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10(5):557–570, 2002. (Cited on pages 4, 15, 21, and 38)

[300] Ewa Syta, Henry Corrigan-Gibbs, Shu-Chun Weng, David Wolinsky, Bryan Ford, and Aaron Johnson. Security Analysis of Accountable Anonymity in Dissent. *ACM Trans. Inf. Syst. Secur.*, 17(1):4:1–4:35, 2014. (Cited on page 152)

[301] The European Commission. Conclusions of the Internet of Things Public Consultation. website, 2013 accessed 2016. http://ec.europa.eu/digital

-agenda/en/news/conclusions-internet-things-public-consultation. (Cited on page 34)

[302] The Internet of Things. website, accessed January 2017. http://www.theinternetofthings.eu/what-is-the-internet-of-things. (Cited on page 30)

[303] Tor Project: Anonymity Online. website, accessed 2017. https://www.torproject.org/. (Cited on page 71)

[304] The Tor project. https://www.torproject.org/, 2003. Accessed Aug 2016. (Cited on pages 24 and 25)

[305] List of Articles Under Tag "Censorship-Circumvention". website, 2016 accessed 2017. https://blog.torproject.org/category/tags/censorship-circumvention. (Cited on page 96)

[306] Tor: Pluggable Transports. website, 2016 accessed 2017. https://www.torproject.org/docs/pluggable-transports.html.en. (Cited on pages 20 and 96)

[307] Rogers Fights BitTorrent by Throttling All Encrypted Transfers. website, 2007 accessed 2017. https://torrentfreak.com/rogers-fighting-bittorrent-by-throttling-all-encrypted-transfers/. (Cited on pages 95 and 96)

[308] Yu-Chuan Tsai, Shyue-Liang Wang, Cheng-Yu Song, and I-Hsien Ting. Privacy and Utility Effects of K-anonymity on Association Rule Hiding. In *Proc. The 3rd Multidisciplinary International Social Networks Conference on SocialInformatics 2016*, MISNC, SI, DS 2016, pages 42:1–42:6, New York, NY, USA, 2016. ACM. (Cited on page 21)

[309] Boris S. Tsybakov and V. A. Mikhailov. Free Synchronous Packet Access in a Broadcast Channel with Feedback. *Problemy Peredachi Informatsii*, 14(4):32–59, 1978. http://www.mathnet.ru/php/getFT.phtml?jrnid=ppi&paperid=1558&what=fullt&option_lang=eng (in Russian). (Cited on page 74)

[310] Ton van Deursen. *50 Ways to Break RFID Privacy*, volume 352 of *IFIP Advances in Information and Communication Technology*, pages 192–205. Springer Berlin Heidelberg, 2011. (Cited on page 36)

[311] Serge Vaudenay. On Privacy Models for RFID. In *Proc. Advances in Cryptology (ASIACRYPT 2007)*, volume 4833 of *Selected Notes in Computer Science*, pages 68–87. Springer-Verlag Berlin Heidelberg, 2007. (Cited on pages 53 and 68)

[312] Patric M. Verrone. Attack of the Killer App. https://en.wikipedia.org/w/index.php?title=Attack_of_the_Killer_App&oldid=760047592, accessed January 2017. Futurama Season 6, Episode 3. (Cited on page 4)

[313] Paul Vines and Tadayoshi Kohno. Rook: Using Video Games As a Low-Bandwidth Censorship Resistant Communication Platform. In *Proc. 14th ACM Workshop on Privacy in the Electronic Society (WPES 2015)*, WPES '15, pages 75–84, New York, NY, USA, 2015. ACM. (Cited on page 20)

[314] Pim Vullers and Gergely Alpár. Efficient Selective Disclosure on Smart Cards Using Idemix. In *Policies and Research in Identity Management: Third IFIP WG 11.6 Working Conference (IDMAN 2013)*, IFIP Advances in Information and Communication Technology, pages 53–67. Springer Berlin Heidelberg, 2013. (Cited on page 35)

[315] Keith Wagstaff. How Target Knew a High School Girl Was Pregnant Before Her Parents Did. website, 2012 (accessed January 2017). http://techland.time.com/2012/02/17/how-target-knew-a-high-school-girl-was-pregnant-before-her-parents/. (Cited on page 2)

[316] Michael Waidner. Unconditional Sender and Recipient Untraceability in Spite of Active Attacks. In Jean-Jacques Quisquater and Joos Vandewalle, editors, *Proc. Advances in Cryptology (EUROCRYPT 1989)*, volume 434 of *Selected Notes in Computer Science*, pages 302–319. Springer Berlin Heidelberg, 1990. (Cited on pages 22, 27, 28, and 74)

[317] Michael Waidner and Birgit Pfitzmann. The Dining Cryptographers in the Disco: Unconditional Sender and Recipient Untraceability with Computationally Secure Serviceability (Abstract). In Jean-Jacques Quisquater and Joos Vandewalle, editors, *Proc. Advances in Cryptology (EUROCRYPT 1989)*, volume 434 of *Selected Notes in Computer Science*, page 690. Springer Berlin Heidelberg, 1990. see also full version [318]. (Cited on page 74)

[318] Michael Waidner and Birgit Pfitzmann. The Dining Cryptographers in the Disco: Unconditional Sender and Recipient Untraceability with Computationally Secure Serviceability. Technical report, Universität Karlsruhe, 1998. (Cited on pages 74, 91, and 197)

[319] Tao Wang and Ian Goldberg. Improved Website Fingerprinting on Tor. In *Proc. 12th ACM Workshop on Privacy in the Electronic Society (WPES)*, WPES '13, pages 201–212, New York, NY, USA, 2013. ACM. (Cited on page 25)

[320] Zachary Weinberg, Jeffrey Wang, Vinod Yegneswaran, Linda Briesemeister, Steven Cheung, Frank Wang, and Dan Boneh. StegoTorus: a Camouflage Proxy for the Tor Anonymity System. In *Proc. 19th ACM Conference on Computer and Communications Security (CCS 2012)*, pages 109–120. ACM, 2012. (Cited on page 104)

[321] Mark Weiser. The Computer for the 21st Century. *Scientific American*, 265(3):66–75, January 1991. (Cited on pages 3 and 29)

[322] Alan Westin. *Privacy and Freedom*. The Bodley Head Ltd, New York, 1970. (Cited on pages 13 and 36)

[323] Reactions to the United States Diplomatic Cables Leak, 2010 accessed 2017. https://en.wikipedia.org/w/index.php?title=Reactions_to_the_United_States_diplomatic_cables_leak&oldid=705141626#Media_outlets. (Cited on page 95)

[324] David Isaac Wolinsky, Henry Corrigan-Gibbs, Bryan Ford, and Aaron Johnson. Dissent in Numbers: Making Strong Anonymity Scale. In *Proc. 10th USENIX Conference on Operating Systems Design and Implementation (OSDI 2012)*, pages 179–192. USENIX Association, 2012. (Cited on pages 25 and 75)

[325] Rebecca Wright and Zhiqiang Yang. Privacy-preserving Bayesian Network Structure Computation on Distributed Heterogeneous Data. In *Proc. Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '04, pages 713–718, New York, NY, USA, 2004. ACM. (Cited on page 21)

[326] Eric Wustrow, Scott Wolchok, Ian Goldberg, and J. Alex Halderman. Telex: Anticensorship in the Network Infrastructure. In *Proc. 20th USENIX Security Symposium (USENIX)*, August 2011. http://www.usenix.org/events/sec/tech/full_papers/Wustrow.pdf. (Cited on pages 20 and 104)

[327] Seokung Yoon, Haeryong Park, and Hyeong Seon Yoo. *Security Issues on Smarthome in IoT Environment*, volume 330 of *Lecture Notes in Electrical Engineering (LNEE)*, pages 691–696. Springer Berlin Heidelberg, 2015. (Cited on page 41)

[328] Adam L. Young and Moti Yung. Kleptography from Standard Assumptions and Applications. In *Security and Cryptography for Networks*, volume 6280 of *Selected Notes in Computer Science*, pages 271–290. Springer Berlin Heidelberg, 2010. (Cited on pages 103 and 106)

[329] Katie Young. 1 in 4 VPN Users Accessing Daily. website, 2016 accessed 2017. http://www.globalwebindex.net/blog/1-in-4-vpn-users-accessing-daily. (Cited on page 95)

[330] Ingmar Zahorsky. Tor, Anonymity, and the Arab Spring: An Interview with Jacob Appelbaum. website, 2011 accessed 2017. http://www.monitor.upeace.org/innerpg.cfm?id_article=816. (Cited on page 96)

[331] Ennan Zhai, David Isaac Wolinsky, Ruichuan Chen, Ewa Syta, Chao Teng, and Bryan Ford. AnonRep: Towards Tracking-Resistant Anonymous Reputation. In *Proc. 13th USENIX Symposium on Networked Systems Design and Implementation (NSDI 16)*, pages 583–596, Santa Clara, CA, 2016. USENIX Association. (Cited on pages xxi and 23)

[332] Justin Zhan, Stan Matwin, and LiWu Chang. Privacy-Preserving Collaborative Association Rule Mining. In *Proc. 19th Annual IFIP WG 11.3 Working Conference on Data and Applications Security*, DBSec'05, pages 153–165, Berlin, Heidelberg, 2005. Springer-Verlag Berlin Heidelberg. (Cited on page 21)

[333] Jan Henrik Ziegeldorf, Fred Grossmann, Martin Henze, Nicolas Inden, and Klaus Wehrle. Coinparty: Secure Multi-party Mixing of Bitcoins. In *Proc. 5th ACM Conference on Data and Application Security and Privacy*, pages 75–86. ACM, 2015. (Cited on page 19)

[334] Jan Henrik Ziegeldorf, Oscar García Morchon, and Klaus Wehrle. Privacy in the Internet of Things: Threats and Challenges. *Security and Communication Networks*, 7(12):2728–2742, December 2014. (Cited on pages 34, 36, and 39)

Abbreviations

AB Attribute-Based.
ABC Attribute-Based Credential.

CCA1 Non-Adaptive Chosen-Ciphertext Attack.
CCA2 Adaptive Chosen-Ciphertext Attack.
CPA Chosen-Plaintext Attack.

DC-net Dining Cryptographers Network.
DDH Decision Diffie-Hellman (assumption).
DH key exchange Diffie-Hellman key exchange.
DoS Denial of Service.

EC Elliptic Curve.
ECC Elliptic-curve Cryptography.

IND Indistinguishability.
IoT Internet of Things.

MAC Message Authentication Code.
Mix-net Mix Network.
MPC Multi-Party Computation.

NFC Near-field communication.

PET Privacy-Enhancing Technology.
PRF Pseudo-Random Function.
PRNG Pseudorandom Number Generator.

RFID Radio Frequency Identification.
RO Random Oracle.

Curriculum Vitae

Anna Krasnova

September 2004 - August 2010 "Specialist degree" in mathematics
(equivalent to a Master's degree)
Computer Security program
Moscow State University of Transport (MIIT), Russia

September 2009 - April 2012 Master of Science
Distributed Systems Engineering program
Technical University of Dresden, Germany

September 2012 - March 2017 PhD
Digital Security, Privacy & Identity Lab
Privacy in the Internet of Things
Radboud University, The Netherlands

April 2017 -
Postdoctoral researcher
Radboud University, The Netherlands

Titles in the IPA Dissertation Series since 2014

J. van den Bos. *Gathering Evidence: Model-Driven Software Engineering in Automated Digital Forensics*. Faculty of Science, UvA. 2014-01

D. Hadziosmanovic. *The Process Matters: Cyber Security in Industrial Control Systems*. Faculty of Electrical Engineering, Mathematics & Computer Science, UT. 2014-02

A.J.P. Jeckmans. *Cryptographically-Enhanced Privacy for Recommender Systems*. Faculty of Electrical Engineering, Mathematics & Computer Science, UT. 2014-03

C.-P. Bezemer. *Performance Optimization of Multi-Tenant Software Systems*. Faculty of Electrical Engineering, Mathematics, and Computer Science, TUD. 2014-04

T.M. Ngo. *Qualitative and Quantitative Information Flow Analysis for Multi-threaded Programs*. Faculty of Electrical Engineering, Mathematics & Computer Science, UT. 2014-05

A.W. Laarman. *Scalable Multi-Core Model Checking*. Faculty of Electrical Engineering, Mathematics & Computer Science, UT. 2014-06

J. Winter. *Coalgebraic Characterizations of Automata-Theoretic Classes*. Faculty of Science, Mathematics and Computer Science, RU. 2014-07

W. Meulemans. *Similarity Measures and Algorithms for Cartographic Schematization*. Faculty of Mathematics and Computer Science, TU/e. 2014-08

A.F.E. Belinfante. *JTorX: Exploring Model-Based Testing*. Faculty of Electrical Engineering, Mathematics & Computer Science, UT. 2014-09

A.P. van der Meer. *Domain Specific Languages and their Type Systems*. Faculty of Mathematics and Computer Science, TU/e. 2014-10

B.N. Vasilescu. *Social Aspects of Collaboration in Online Software Communities*. Faculty of Mathematics and Computer Science, TU/e. 2014-11

F.D. Aarts. *Tomte: Bridging the Gap between Active Learning and Real-World Systems*. Faculty of Science, Mathematics and Computer Science, RU. 2014-12

N. Noroozi. *Improving Input-Output Conformance Testing Theories*. Faculty of Mathematics and Computer Science, TU/e. 2014-13

M. Helvensteijn. *Abstract Delta Modeling: Software Product Lines and Beyond*. Faculty of Mathematics and Natural Sciences, UL. 2014-14

P. Vullers. *Efficient Implementations of Attribute-based Credentials on Smart Cards*. Faculty of Science, Mathematics and Computer Science, RU. 2014-15

F.W. Takes. *Algorithms for Analyzing and Mining Real-World Graphs*. Faculty of Mathematics and Natural Sciences, UL. 2014-16

M.P. Schraagen. *Aspects of Record Linkage*. Faculty of Mathematics and Natural Sciences, UL. 2014-17

G. Alpár. *Attribute-Based Identity Management: Bridging the Cryptographic Design of ABCs with the Real World*. Faculty of Science, Mathematics and Computer Science, RU. 2015-01

A.J. van der Ploeg. *Efficient Abstractions for Visualization and Interaction*. Faculty of Science, UvA. 2015-02

R.J.M. Theunissen. *Supervisory Control in Health Care Systems*. Faculty of Mechanical Engineering, TU/e. 2015-03

T.V. Bui. *A Software Architecture for Body Area Sensor Networks: Flexibility and Trustworthiness*. Faculty of Mathematics and Computer Science, TU/e. 2015-04

A. Guzzi. *Supporting Developers' Teamwork from within the IDE*. Faculty of Electrical Engineering, Mathematics, and Computer Science, TUD. 2015-05

T. Espinha. *Web Service Growing Pains: Understanding Services and Their Clients*. Faculty of Electrical Engineering, Mathematics, and Computer Science, TUD. 2015-06

S. Dietzel. *Resilient In-network Aggregation for Vehicular Networks*. Faculty of Electrical Engineering, Mathematics & Computer Science, UT. 2015-07

E. Costante. *Privacy throughout the Data Cycle*. Faculty of Mathematics and Computer Science, TU/e. 2015-08

S. Cranen. *Getting the point — Obtaining and understanding fixpoints in model checking*. Faculty of Mathematics and Computer Science, TU/e. 2015-09

R. Verdult. *The (in)security of proprietary cryptography*. Faculty of Science, Mathematics and Computer Science, RU. 2015-10

J.E.J. de Ruiter. *Lessons learned in the analysis of the EMV and TLS security protocols*. Faculty of Science, Mathematics and Computer Science, RU. 2015-11

Y. Dajsuren. *On the Design of an Architecture Framework and Quality Evaluation for Automotive Software Systems*. Faculty of Mathematics and Computer Science, TU/e. 2015-12

J. Bransen. *On the Incremental Evaluation of Higher-Order Attribute Grammars*. Faculty of Science, UU. 2015-13

S. Picek. *Applications of Evolutionary Computation to Cryptology*. Faculty of Science, Mathematics and Computer Science, RU. 2015-14

C. Chen. *Automated Fault Localization for Service-Oriented Software Systems*. Faculty of Electrical Engineering, Mathematics, and Computer Science, TUD. 2015-15

S. te Brinke. *Developing Energy-Aware Software*. Faculty of Electrical Engineering, Mathematics & Computer Science, UT. 2015-16

R.W.J. Kersten. *Software Analysis Methods for Resource-Sensitive Systems*. Faculty of Science, Mathematics and Computer Science, RU. 2015-17

J.C. Rot. *Enhanced coinduction*. Faculty of Mathematics and Natural Sciences, UL. 2015-18

M. Stolikj. *Building Blocks for the Internet of Things*. Faculty of Mathematics and Computer Science, TU/e. 2015-19

D. Gebler. *Robust SOS Specifications of Probabilistic Processes*. Faculty of Sciences, Department of Computer Science, VUA. 2015-20

M. Zaharieva-Stojanovski. *Closer to Reliable Software: Verifying functional behaviour of concurrent programs*. Faculty of Electrical Engineering, Mathematics & Computer Science, UT. 2015-21

R.J. Krebbers. *The C standard formalized in Coq*. Faculty of Science, Mathematics and Computer Science, RU. 2015-22

R. van Vliet. *DNA Expressions – A Formal Notation for DNA*. Faculty of Mathematics and Natural Sciences, UL. 2015-23

S.-S.T.Q. Jongmans. *Automata-Theoretic Protocol Programming*. Faculty of Mathematics and Natural Sciences, UL. 2016-01

S.J.C. Joosten. *Verification of Interconnects*. Faculty of Mathematics and Computer Science, TU/e. 2016-02

M.W. Gazda. *Fixpoint Logic, Games, and Relations of Consequence*. Faculty of Mathematics and Computer Science, TU/e. 2016-03

S. Keshishzadeh. *Formal Analysis and Verification of Embedded Systems for Healthcare*. Faculty of Mathematics and Computer Science, TU/e. 2016-04

P.M. Heck. *Quality of Just-in-Time Requirements: Just-Enough and Just-in-Time*. Faculty of Electrical Engineering, Mathematics, and Computer Science, TUD. 2016-05

Y. Luo. *From Conceptual Models to Safety Assurance – Applying Model-Based Techniques to Support Safety Assurance*. Faculty of Mathematics and Computer Science, TU/e. 2016-06

B. Ege. *Physical Security Analysis of Embedded Devices*. Faculty of Science, Mathematics and Computer Science, RU. 2016-07

A.I. van Goethem. *Algorithms for Curved Schematization*. Faculty of Mathematics and Computer Science, TU/e. 2016-08

T. van Dijk. *Sylvan: Multi-core Decision Diagrams*. Faculty of Electrical Engineering, Mathematics & Computer Science, UT. 2016-09

I. David. *Run-time resource management for component-based systems*. Faculty of Mathematics and Computer Science, TU/e. 2016-10

A.C. van Hulst. *Control Synthesis using Modal Logic and Partial Bisimilarity – A Treatise Supported by Computer Verified Proofs*. Faculty of Mechanical Engineering, TU/e. 2016-11

A. Zawedde. *Modeling the Dynamics of Requirements Process Improvement*. Faculty of Mathematics and Computer Science, TU/e. 2016-12

F.M.J. van den Broek. *Mobile Communication Security*. Faculty of Science, Mathematics and Computer Science, RU. 2016-13

J.N. van Rijn. *Massively Collaborative Machine Learning*. Faculty of Mathematics and Natural Sciences, UL. 2016-14

M.J. Steindorfer. *Efficient Immutable Collections*. Faculty of Science, UvA. 2017-01

W. Ahmad. *Green Computing: Efficient Energy Management of Multiprocessor Streaming Applications via Model Checking*. Faculty of Electrical Engineering, Mathematics & Computer Science, UT. 2017-02

D. Guck. *Reliable Systems – Fault tree analysis via Markov reward automata*. Faculty of Electrical Engineering, Mathematics & Computer Science, UT. 2017-03

H.L. Salunkhe. *Modeling and Buffer Analysis of Real-time Streaming Radio Applications Scheduled on Heterogeneous Multiprocessors*. Faculty of Mathematics and Computer Science, TU/e. 2017-04

A. Krasnova. *Smart invaders of private matters: Privacy of communication on the Internet and in the Internet of Things (IoT)*. Faculty of Science, Mathematics and Computer Science, RU. 2017-05